

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ЕЁ ПРИМЕНЕНИЕ

Новое
в жизни,
науке,
технике

Подписная
научно-
популярная
серия

Издается
ежемесячно
с 1988 г.

Графика в байтах

1991

10

Новое
в жизни,
науке,
технике

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ЕЁ ПРИМЕНЕНИЕ

Подписная
научно-
популярная
серия

10/1991

Издается
ежемесячно
с 1988 г.

ГРАФИКА В БАЙТАХ

В номере:



Москва
Издательство
"Знание"
1991

ББК 32.97
Г 27



ПРОКОПЕНКО АЛЕКСАНДР АНДРЕЕВИЧ — ведущий инженер НИИ связи, специалист в области машинной графики.
ЖАРИКОВ ЛЕОНИД НИКОЛАЕВИЧ — научный сотрудник, занимается разработкой программного обеспечения АСУ.
НОСКОВА НАТАЛЬЯ БОРИСОВНА — инженер-программист.
ПЕРЕХОД ИГОРЬ АЛЕКСАНДРОВИЧ — кандидат физико-математических наук, доцент Симферопольского государственного университета.
УСЕНКОВ ДМИТРИЙ ЮРЬЕВИЧ — студент МГТУ им. Н.Э.Баумана, программист.

Редактор Б.М.Васильев

ISBN 5-07-002202-4



Представляем вам популярную во всем мире графическую систему AutoCAD, работающую на персональных компьютерах семейства IBM PC или им подобных. Предполагается, что читатель имеет элементарные представления о работе на этих компьютерах под управлением операционной системы MS-DOS.

А. А. Прокопенко

ГРАФИЧЕСКАЯ СИСТЕМА AUTOCAD

Введение

Советскому читателю уже знакома графическая система AutoCAD, продукт фирмы Autodesk, по книге [1]. Однако, несмотря на свой большой тираж (30 тыс. экз.), эта книга тут же стала библиографической редкостью. Кроме того, в ней описана достаточно устаревшая версия 2.17. В настоящее время в эксплуатации находится версия 10, завоевавшая в 1989 году международный приз за лучшую научно-техническую программу года. Имеется и русифицированный вариант этой версии, разработанный совместным советско-британским предприятием "ПАРАЛЛЕЛЬ". Работы по совершенствованию системы AutoCAD продолжаются: есть сведения о выпуске 11-й англоязычной версии.

Статья не претендует на справочное пособие по системе AutoCAD, она преследует другую цель: в доступной форме показать читателю, не избалованному специальной литературой по AutoCADу, некоторые возможности системы. Остановим свой выбор на англоязычной версии 10. Кто имеет возможность приобрести в СП "ПАРАЛЛЕЛЬ" аналогичную русифицированную версию, будет ориентироваться на подстрочный перевод английского текста, с которым здесь встретится.

Что может AutoCAD

AutoCAD - это мощная, открытая для развития система, способная автоматизировать самые разнообразные графические работы. Ограничений здесь практически никаких. С помощью AutoCADa можно создавать машиностроительные чертежи, графическую документацию для выпуска радиоэлектронной аппаратуры, архитектурно-строительные чертежи, чертежи для судостроительной и авиационной промышленности, картографическую документацию, технические и художественные иллюстрации (в том числе и цветные) и даже мультфильмы. Высокая производительность системы достигается как ее собственными программными средствами, так и конструкциями пользователя, которые он описывает специальным языком AutoLISP, понятным системе AutoCAD.

Создаем свой первый чертеж

Итак, вы приобрели AutoCAD и настроили его на имеющиеся у вас технические средства. Желательно, чтобы, помимо плоттера, с помощью которого вы получите готовый чертеж на бумаге, в комплект вашей ЭВМ входило устройство дистанционного управления типа "мышь" - оно сделает работу с AutoCADом не только приятным занятием, но и существенно ее ускорит.

Поставим перед собой цель: выпустим с помощью AutoCADa чертеж детали, изображенной на рис. 1. Будем идти к намеченной цели по методу KEY-BY-KEY (клавиша за клавишей).

После запуска AutoCADa (имя основной программы - ACAD) на экране монитора появится главное меню, изображенное на рис. 2.

Поскольку чертеж (рисунок) создается впервые, следует выбрать первый пункт меню. Сообщим о своем решении AutoCADу, нажав клавиши "1" и <CR>. Вместо <CR> (Carriage Return - Возврат Каретки) или <Return>

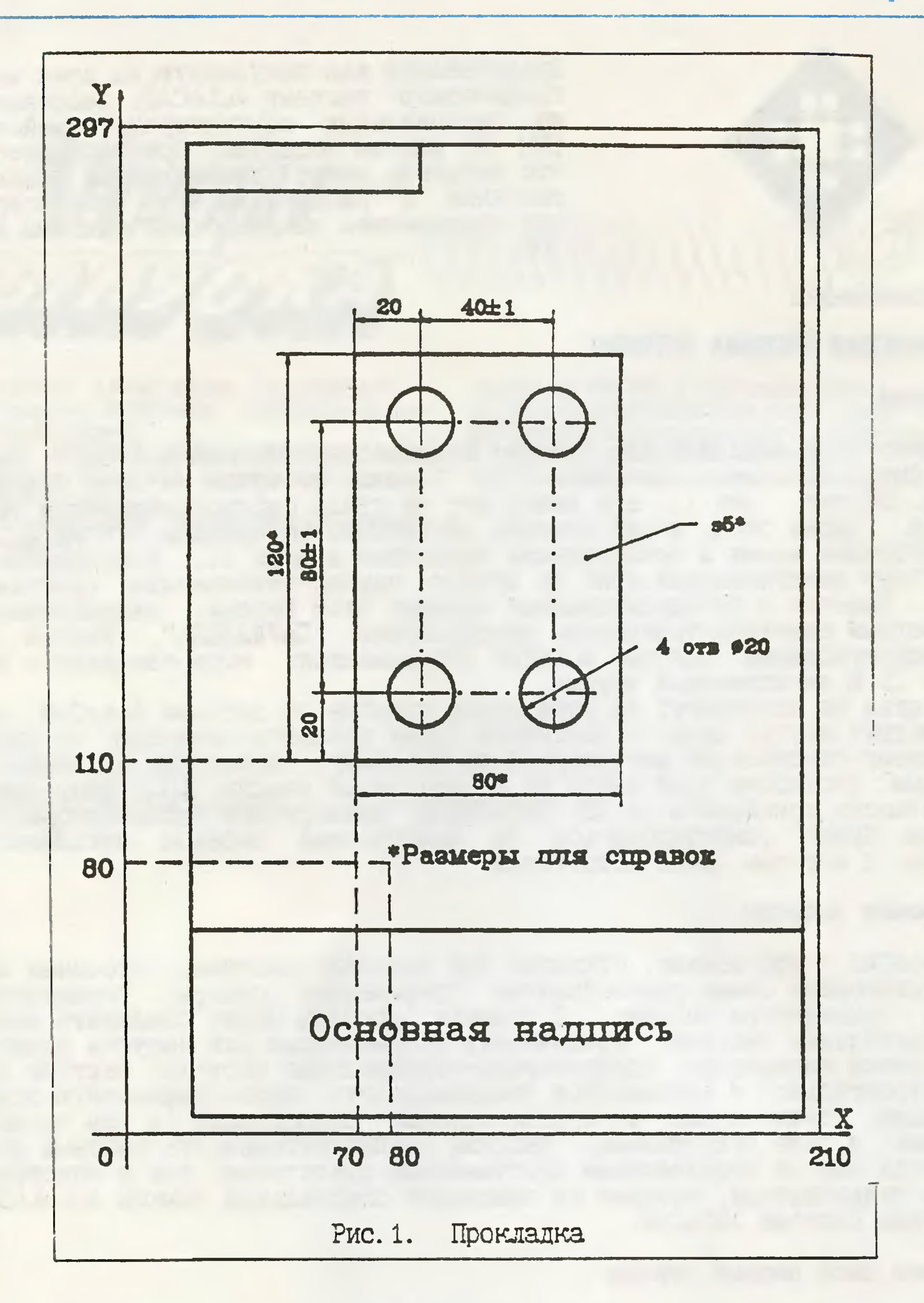


Рис. 1. Прокладка

(Возврат) можно пользоваться клавишей <Enter> (Ввод). Любая из этих клавишей завершает ввод информации в AutoCAD:

Enter selection: ***1<CR>***
Ваш выбор: ***1<CR>***

На экране монитора появится запрос имени рисунка (чтобы сохранить рисунок под этим именем в конце сеанса). Имя может содержать от одного до восьми символов. Здесь разрешается использовать лишь латинские буквы, цифры, $ (доллар), дефис и знак подчеркивания, но в любом сочетании.

```
Main Menu
Главное меню

    0. Exit AutoCAD
    0. Выйти из Автокада

    1. Begin a NEW drawing
    1. Начать НОВЫЙ рисунок

    2. Edit an EXISTING drawing
    2. Отредактировать СУЩЕСТВУЮЩИЙ рисунок

    3. Plot a drawing
    3. Вычертить рисунок на плоттере

    4. Printer Plot a drawing
    4. Распечатать рисунок на принтере

    5. Configure AutoCAD
    5. Настроить Автокад

    6. File Utilities
    6. Работа с файлами

    7. Compile shape/font description file
    7. Компиляция файла форм или шрифтов

    8. Convert old drawing file
    8. Обновление рисунка, созданного старой версией Автокада

Enter selection:
Ваш выбор:
```

Рис. 2. Главное меню AutoCADa
(экран монитора в текстовом режиме)

Присвоим нашему будущему чертежу имя PROKLAD, сообщив об этом AutoCADу:

Enter NAME of drawing: ***PROKLAD<CR>***
Введите ИМЯ рисунка: ***PROKLAD<CR>***

После этого AutoCAD автоматически перейдет из текстового режима в графический. Экран монитора при этом будет разбит на четыре зоны, как это показано на рис. 3. В текстовой зоне (нижняя часть экрана) AutoCAD сформирует запрос:

Command:
Команда:

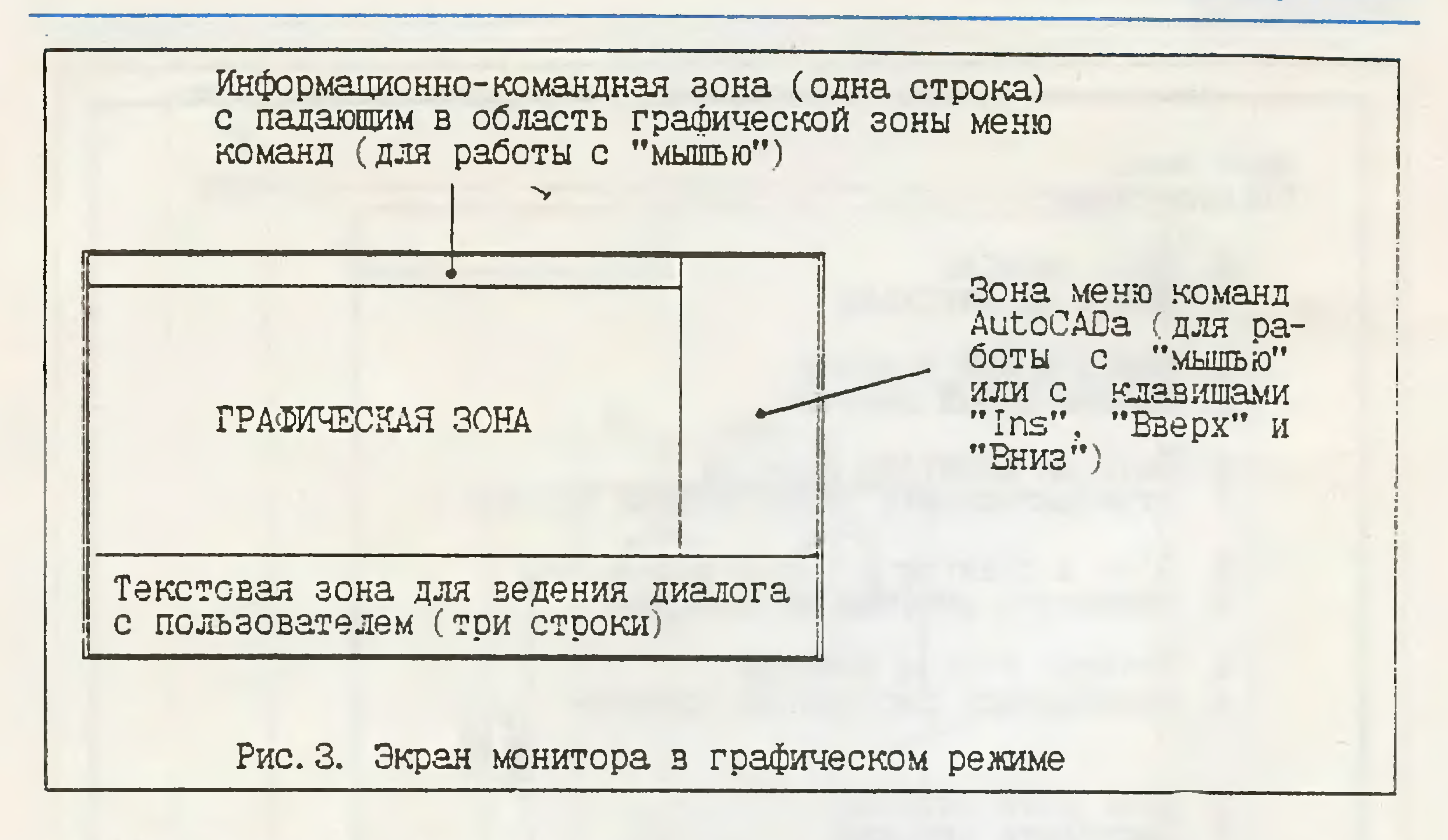


Рис. 3. Экран монитора в графическом режиме

С этого момента путем задания AutoCADу различных команд (как с клавиатуры, так и "мышью") пользователь может формировать чертеж. Распишем наши действия по шагам.

1. Прежде всего, зададим AutoCADу формат будущего чертежа. Для нашего случая это будет формат А4 размером 210x297 мм:

Command: ***LIMITS<CR>***
Команда: ***ЛИМИТЫ<CR>***

После этого последует запрос координат левого нижнего угла формата, в угловых скобках AutoCAD предложит нам свой вариант ответа на свой запрос (не обязательно такой, как указано здесь). Нажав клавишу <CR> (ответ по умолчанию), мы соглашаемся с предложенными AutoCADом координатами:

ON/OFF/<Lower left corner><0.0000,0.0000>: ***<CR>***
Вкл/Откл/<Левый нижний угол><0.0000,0.0000>: ***<CR>***

Здесь через запятую идут координаты X и Y в миллиметрах, точкой отделяется целая часть числа от дробной.

Далее AutoCAD запрашивает координаты верхнего правого угла. Вариант его ответа нас уже не устраивает, и мы вводим свой:

Upper right corner <12.0000,9.0000>: ***210,297<CR>***
Верхний правый угол <12.0000,9.0000>: ***210,297<CR>***

2. Сделаем так, чтобы весь формат целиком уместился на экране монитора:

Command: ***ZOOM<CR>***
Команда: ***ПОКАЖИ<CR>***

All/Center/...: ***ALL<CR>***
Все/Центр/...: ***ВСЕ<CR>***

Уместно отметить, что для ускорения ввода информации достаточно вводить лишь ту часть слова, выбранного из меню ответов, которая выделена прописными буквами. Впредь будем иметь это в виду.

3. Обратим внимание на то, что все размеры нашей детали и точка привязки ее нижней левой точки к полю чертежа кратны 10 мм (см. рис. 1). Зададим это значение в качестве разрешающей способности AutoCADa, тогда при использовании "мыши" или клавиш перемещения мар-

кера координаты всех вводимых точек будут автоматически втягиваться в дискрет 10 мм:

```
Command: SNAP<CR>
Команда: ШАГ<CR>

Snap spacing or ON/OFF/Aspect/Rotate/Style <1.0000>: 10<CR>
Шаг привязки или Вкл/Откл/Аспект/Поворот/Стиль <1.0000>: 10<CR>
```

4. Включим режим вывода на экран координатной сетки, узлы которой будут отстоять друг от друга на величину введенной нами разрешающей способности:

```
Command: GRID<CR>
Команда: СЕТКА<CR>

Grid spacing(X) or ON/OFF/Snap/Aspect <1.0000>: 10<CR>
Интервал сетки(X) или Вкл/Откл/Шаг/Аспект <1.0000>: 10<CR>
```

Сетка покроет не весь экран монитора, а лишь область заданного нами в шаге 1 формата. Теперь мы видим границы своего листа, к тому же AutoCAD будет сам следить за тем, чтобы все вводимые нами координаты (с клавиатуры или с помощью "мыши") не выходили за пределы нашего формата.

5. Используя стандартный набор элементарных графических образов или примитивов (ломаная линия или полилиния, текстовая строка и др.) сформируем чертеж детали. Будем считать, что мы располагаем дешевым одноперьевым плоттером с пером в 0.25 мм. Поэтому для вычерчивания элементов чертежа сплошной основной линией шириной, скажем, 0.6 мм потребуется многопроходной режим работы плоттера. Такой режим способен обеспечить примитив PLINE (ПЛИНИЯ). Этот примитив, будучи раз настроенным на определенную ширину линии, будет "помнить" ее до конца сеанса работы с AutoCADом или до очередной перенастройки.

Начнем с прорисовки контура детали и обойдем его по часовой стрелке от нижнего левого угла:

```
Command: PLINE<CR>
Команда: ПЛИНИЯ<CR>

From point: 70,110<CR>
От точки: 70,110<CR>
```

Настроим примитив PLINE (ПЛИНИЯ) на ширину линии в 0.6 мм:

```
Arc/Close/.../Width/<Endpoint of line>: W<CR>
Дуга/Замкни/.../Ширина/<Конечная точка сегмента>: Ш<CR>

Starting width <0.0000>: 0.6<CR>
Начальная ширина <0.0000>: 0.6<CR>
```

После загрузки начальной ширины линии AutoCAD попросит указать конечную ширину линии. Не удивляйтесь этому - AutoCAD позволяет изображать и сужающиеся или расширяющиеся линии:

```
Ending width <0.6000>: <CR>
Конечная ширина <0.6000>: <CR>

Arc/Close/.../Width/<Endpoint of line>: @120<90<CR>
Дуга/Замкни/.../Ширина/<Конечная точка сегмента>: @120<90<CR>
```

Подобная запись означает: провести отрезок длиною 120 мм от предыдущей точки (в нашем случае от точки с координатами {70,110}) под углом 90 градусов к линии горизонта. Допустима и координатная форма записи конечной точки отрезка, т.е. {70,230}. Укажем положение и других вершин прямоугольника:

Arc/Close/.../Width/<Endpoint of line>: ***@80<0<CR>***
ДУга/Замкни/.../Ширина/<Конечная точка сегмента>: ***@80<0<CR>***

Arc/Close/.../Width/<Endpoint of line>: ***@120<270<CR>***
ДУга/Замкни/.../Ширина/<Конечная точка сегмента>: ***@120<270<CR>***

Если при наборе была допущена ошибка, а информация отправлена на обработку (нажатием <CR>), неправильную линию тут же можно удалить, набрав "U" ("ОТМ") и нажав <CR>. Таким же способом можно исправить ошибочные действия оператора не только внутри команды, но и отменить всю неверную и уже обработанную AutoCADом команду. Более того, многократный повтор отмены позволяет ликвидировать одну за другой последние введенные команды вплоть до первой (в нашем случае - команды LIMITS) и в конце концов до появления предупреждения "Nothing to undo" ("Нечего отменять").

Замкнем контур, подав указание Close (Замкни):

Arc/Close/.../Width/<Endpoint of line>: ***C<CR>***
ДУга/Замкни/.../Ширина/<Конечная точка сегмента>: ***З<CR>***

6. Для удобства работы увеличим изображение прямоугольника на экране:

Command: ***ZOOM<CR>***
Команда: ***ПОКАЖИ<CR>***

All/Center/.../Window/...: ***W<CR>***
Все/Центр/.../Рамка/...: ***Р<CR>***

First corner: ***40,90<CR>***
Первый угол: ***40,90<CR>***

Other corner: ***210,250<CR>***
Другой угол: ***210,250<CR>***

7. Изобразим левое нижнее отверстие. Поскольку окружность следует вычерчивать линией той же ширины, что и при прорисовке контура детали, вновь воспользуемся примитивом PLINE (ПЛИНИЯ). Однако окружность придется рисовать фрагментами, т.е. дугами:

Command: ***PLINE<CR>***
Команда: ***ПЛИНИЯ<CR>***

Введем координаты крайней правой точки окружности:

From point: ***100,130<CR>***
От точки: ***100,130<CR>***

Зададим режим изображения дуги (полуокружности) и ее параметры:

Arc/Close/.../Width/<Endpoint of line>: ***A<CR>***
ДУга/Замкни/.../Ширина/<Конечная точка сегмента>: ***ДУ<CR>***

Angle/CEnter/CLose/...<Endpoint of arc>: ***CE<CR>***
Угол/Центр/Замкни/...<Конечная точка дуги>: ***Ц<CR>***

Center point: ***90,130<CR>***
Центр: ***90,130<CR>***

Angle/Length/<End point>: ***A<CR>***
Угол/Длина/<Конечная точка>: ***У<CR>***

Included angle: ***180<CR>***
Центральный угол: ***180<CR>***

Замкнем дугу по кругу:

```
Angle/CEnter/CLose/...<Endpoint of arc>: CL<CR>
Угол/Центр/Замкни/...<Конечная точка дуги>: З<CR>
```

8. Размножим полученную окружность до четырех:

```
Command: ARRAY<CR>
Команда: МАССИВ<CR>
```

Для размножения выберем *Последний (Last)* сформированный нами объект, т.е. окружность:

```
Select objects: L<CR>
Выберите объекты: П<CR>
```

AutoCAD сообщит о количестве выделенных и подсвеченных им объектов и предложит нам продолжить выбор размножаемых объектов:

```
1 found.
Select objects: <CR>
1 найден(ы).
Выберите объекты: <CR>
```

После отказа (по умолчанию) от предложения AutoCADа зададим ему способ размножения выделенного объекта:

```
Rectangular or Polar array (R/P): R<CR>
Прямоугольный или Круговой массив (П/К): П<CR>
```

Зададим размер прямоугольного массива (в количестве окружностей по вертикали и горизонтали) и дистанцию между элементами массива:

```
Number of rows (---)<1>: 2<CR>
Число строк (---)<1>: 2<CR>

Number columns (|||)<1>: 2<CR>
Число столбцов (|||)<1>: 2<CR>

Unit cell or distance between rows (---): 80<CR>
Размер ячейки или расстояние между строками (---): 80<CR>

Distance between columns (|||): 40<CR>
Расстояние между столбцами (|||): 40<CR>
```

9. Изобразим штрих-пунктирные линии.
Загрузим нужный тип линии (штрих-пунктирный), его имя - DASHDOT:

```
Command: LINETYPE<CR>
Команда: ТИПЛИН<CR>

?/Create/Load/Set: L<CR>
?/Создай/Загрузи/Установи: З<CR>

Linetype(s) to load: DASHDOT<CR>
Загрузить тип линии: DASHDOT<CR>

File to searh <ACAD>: <CR>
Искать в файле <ACAD>: <CR>

Linetype DASHDOT loaded.
Тип линии DASHDOT загружен.
```

Установим загруженный нами тип линии, т.е. сделаем его текущим:

?/Create/Load/Set: ***S<CR>***
?/Создай/Загрузи/Установи: ***У<CR>***

New entity linetype (or ?)<BYLAYER>: ***DASHDOT<CR>***
Новый тип линии (или ?)<ПОСЛОЮ>: ***DASHDOT<CR>***

Покинем команду LINETYPE (ТИПЛИН):

?/Create/Load/Set: ***<CR>***
?/Создай/Загрузи/Установи: ***<CR>***

Проведем верхнюю горизонтальную штрих-пунктирную линию:

Command: ***LINE<CR>***
Команда: ***ОТРЕЗОК<CR>***

From point: ***90,210<CR>***
От точки: ***90,210<CR>***

Координаты второй точки зададим в приращениях по отношению к первой точке:

To point: ***@52,0<CR>***
К точке: ***@52,0<CR>***

Завершим формирование отрезка:

To point: ***<CR>***
К точке: ***<CR>***

Может случиться, что полученные штрихи и пробелы между ними оказались столь малы, что линия кажется сплошной или, наоборот, штрихи и пробелы слишком велики. В этом случае следует воспользоваться командой LTSCALE (ЛМАСШТАБ) и исправить дефект:

Command: ***LTSCALE<CR>***
Команда: ***ЛМАСШТАБ<CR>***

New scale factor <1.0000>: ***25<CR>***
Новый масштаб <1.0000>: ***25<CR>***

Если вас вновь не устроило изображение штрих-пунктирной линии, нажмите <CR> и введите новый масштабный коэффициент.

Проведем нижнюю горизонтальную штрих-пунктирную линию:

Command: ***LINE<CR>***
Команда: ***ОТРЕЗОК<CR>***

From point: ***90,130<CR>***
От точки: ***90,130<CR>***

To point: ***@52,0<CR>***
К точке: ***@52,0<CR>***

To point: ***<CR>***
К точке: ***<CR>***

Проведем левую вертикальную штрих-пунктирную линию. Так как мы будем использовать ту же команду LINE (ОТРЕЗОК), то на запрос AutoCADа о вводе новой команды достаточно нажать <CR>. Это означает для AutoCADа: следующая команда будет такой же, как и предыдущая. Не пренебрегайте системным сервисом, он сэкономит время вашей работы.

```
Command: <CR>
Команда: <CR>

LINE From point: 90,210<CR>
ОТРЕЗОК От точки: 90,210<CR>

To point: @0,-92<CR>
К точке: @0,-92<CR>

To point: <CR>
К точке: <CR>
```

Проведем правую вертикальную штрих-пунктирную линию:

```
Command: <CR>
Команда: <CR>

LINE From point: 130,210<CR>
ОТРЕЗОК От точки: 130,210<CR>

To point: @0,-92<CR>
К точке: @0,-92<CR>

To point: <CR>
К точке: <CR>
```

Восстановим сплошную основную линию (тип линии - CONTINUOUS):

```
Command: LINETYPE<CR>
Команда: ТИПЛИН<CR>

?/Create/Load/Set: S<CR>
?/Создай/Загрузи/Установи: У<CR>

New entity linetype (or ?)<DASHDOT>: CONTINUOUS<CR>
Новый тип линии (или ?)<DASHDOT>: CONTINUOUS<CR>
```

Покинем команду LINETYPE (ТИПЛИН):

```
?/Create/Load/Set: <CR>
?/Создай/Загрузи/Установи: <CR>
```

Если вам мешают вспомогательные засечки, которые AutoCAD наносит на экране при проведении графических построений, их в любое время можно удалить командой REDRAW (ОСВЕЖИ).

10. Подготовимся к простановке размеров.

Обратим внимание на то, что все размеры (в том числе и допуски) - целые числа. Сообщим об этом AutoCADу:

```
Command: UNITS<CR>
Команда: ЕДИНИЦЫ<CR>

Systems of units:           (Examples)
  1. Scientific               1.55E+01
  2. Decimal                  15.50
  3. Engineering              1'-3.50"
  4. Architectural            1'-3 1/2"
  5. Fractional               15 1/2

Enter choice, 1 to 5 <2>: <CR>

Единицы измерения:          (Примеры)
  1. Научные                  1.55E+01
  2. Десятичные               15.50
  3. Технические              1'-3.50"
```

```
  4. Архитектурные             1'-3 1/2"
  5. С дробной частью          15 1/2
        .   .   .
  Ваш выбор - от 1 до 5 <2>: <CR>

Number of digits to right of decimal point (0 to 8) <4>: 0<CR>
Число знаков после запятой (от 0 до 8) <4>: 0<CR>
```

На последующие вопросы системы отвечаем нажатием <CR>, пока не появится запрос о вводе новой команды.

Вы, конечно же, обратили внимание на то, что при использовании команды UNITS (ЕДИНИЦЫ) AutoCAD самостоятельно перешел из графического режима в текстовый (иначе не поместилось бы меню). Не будем спешить с обратным возвратом в графический режим. Поскольку вы работаете с AutoCADом впервые, поинтересуемся, как настроены переменные параметры размерных линий. Для этого введите (строчные буквы можно опустить):

```
Command: DIM<CR>
Команда: РАЗМЕР<CR>

Dim: STAtus<CR>
Размер: СТАтус<CR>
```

После этого на экране появится список параметров размерных линий с указанием их значений (нажав клавишу <CR>, можно просмотреть список до конца). Обратим внимание на некоторые из этих параметров:

```
DIMASZ      3      Arrow size
РЗМВЛСТ     3      Величина стрелки

DIMEXE      3      Extension above dimension line
РЗМПВЛ      3      Длина выноски над размерной линией

DIMTAD      On     Place text above the dimension line
РЗМТНРЛ     Вкл    Поместить текст над размерной линией

DIMTIH      Off    Text inside extensions is horizontal
РЗМТМЕЖГ    Откл   Текст между размерными линиями горизонтален

DIMTM       1      Minus tolerance
РЗМДМИН     1      Отрицательный допуск

DIMTOH      Off    Text outside extensions is horizontal
РЗМТВНЕГ    Откл   Текст за размерными линиями горизонтален

DIMTOL      Off    Generate dimension tolerances
РЗМДОП      Откл   Генерация размерных допусков

DIMTP       1      Plus tolerance
РЗМДПЛ      1      Положительный допуск

DIMTXT      4      Text height
РЗМТЕКСТ    4      Высота текста
```

Если на своем экране вы найдете параметры, отличающиеся от указанных, скорректируйте их. Например, параметр DIMTM (РЗМДМИН) оказался со значением "0". Изменим его на "1":

```
Dim: DIMTM<CR>
Размер: РЗМДМИН<CR>

Current value <0>  New value: 1<CR>
Текущее значение <0>  Новое значение: 1<CR>
```

Для выхода на запрос ввода команды нажмите <Ctrl-C> или введите:

Dim: ***EXIT<CR>***
Размер: ***ВЫХОД<CR>***

Чтобы возвратиться в экранный режим, введите команду REDRAW (ОСВЕЖИ). В том случае если вы пользуетесь "мышью", для восстановления зоны меню переведите маркер в правый верхний угол экрана монитора и нажмите "мышиную" клавишу ввода информации; если работаете с клавишами "Ins", "Вверх", "Вниз", нажмите "Ins" и, подняв светящийся прямоугольник в верхний правый угол, повторно нажмите "Ins".

При простановке размеров потребуется использование специальных символов, отсутствующих на клавиатуре: знак допуска и знак диаметра. Как быть в этом случае? AutoCAD имеет в своем арсенале множество шрифтов (более 30), среди которых шрифт CYRILTLC содержит нужные нам символы. Нажатие заглавной русской буквы "А" вызовет формирование знака допуска, заглавной "Б" - знака диаметра. Для набора фразы "отв" следует набрать в латинском регистре "otv", в отношении цифр исключений нет - они вводятся обычным способом. Если же потребуется ввод более обширного русского текста (например, при формировании технических требований), шрифт CYRILTLC будет уже неудобен, его следует заменить шрифтом CYRILL, специально предназначенным для ввода русских и латинских букв. Поскольку нам потребуются оба шрифта, целесообразно их настроить сразу - в этом случае процедура смены шрифтов намного ускорится. В поставляемой англоязычной версии AutoCADa заложен шрифт STANDARD (в русскоязычной - СТАНДАРТ), содержащий одновременно русский и латинский алфавиты, но конфигурация шрифта очень упрощена.

Начнем настройку шрифта CYRILL:

Command: ***STYLE<CR>***
Команда: ***СТИЛЬ<CR>***

Text style name (or ?) <STANDARD>: ***CYRILL<CR>***
Имя гарнитуры шрифта (или ?) <СТАНДАРТ>: ***CYRILL<CR>***

New style.
Font file <txt>: ***CYRILL<CR>***
Новая гарнитура.
Файл шрифта <txt>: ***CYRILL<CR>***

Height <0>: ***<CR>***
Высота <0>: ***<CR>***

На остальные запросы отвечаем нажатием <CR> вплоть до появления запроса на ввод команды. Вы, конечно, обратили внимание, что на запрос Height (Высота) мы ответили нажатием <CR>. Это означает, что мы отказались от фиксированной настройки высоты шрифта CYRILL и будем задавать этот параметр при непосредственном использовании шрифта. Действительно, при заполнении основной надписи потребуются шрифты одной и той же конфигурации или гарнитуры, но различной высоты. При настройке же шрифта CYRILTLC укажем и его высоту: 4 мм. Настройте этот шрифт самостоятельно так же, как мы настраивали CYRILL.

11. Так как последним был настроен шрифт CYRILTLC, то с ним и поработаем. Проставим размер на отверстии:

Command: ***DIM<CR>***
Команда: ***РАЗМЕР<CR>***

Dim: ***DIAM<CR>***
Размер: ***ДИАметр<CR>***

Select arc or circle: ***139,135<CR>***
Выберите дугу или круг: ***139,135<CR>***

Dimension text <20>: ***4 otv Б20<CR>***
Размерный текст <20>: ***4 otv Б20<CR>***

В англоязычной версии вы возможно, и не увидите букву "Б" в нижней части экрана, тем не менее знак диаметра будет сформирован в графической зоне экрана после указания длины выноски:

Text does not fit.
Enter leader length for text: ***20<CR>***
Текст не помещается.
Введите длину выноски для текста: ***20<CR>***

12. Проставим верхние горизонтальные размеры. Проставим удаление центра отверстия от левой кромки детали (20 мм):

Dim: ***HOR<CR>***
Размер: ***ГОР<CR>***

First extension line origin or RETURN to select: ***70,230<CR>***
Начало первой выносной линии или RETURN-для выбора: ***70,230<CR>***

Second extension line origin: ***@20,-20<CR>***
Начало второй выносной линии: ***@20,-20<CR>***

Dimension line location: ***70,240<CR>***
Место положения размерной линии: ***70,240<CR>***

Dimension text <20>: ***<CR>***
Размерный текст <20>: ***<CR>***

Проставим горизонтальное межцентровое расстояние (40 мм). Включим простановку допуска:

Dim: ***DIMTOL<CR>***
Размер: ***РЗМДОП<CR>***

Current value <Off> New value: ***ON<CR>***
Текущее значение <Откл> Новое значение: ***Вкл<CR>***

Разрешим AutoCADу проставить размер с допуском самостоятельно:

Dim: ***CONT<CR>***
Размер: ***ПРОдолжение<CR>***

Введем координаты центра правого верхнего отверстия:

Second extension line origin: ***130,210<CR>***
Начало второй выносной линии: ***130,210<CR>***

Dimension text <40>: ***<CR>***
Размерный текст <40>: ***<CR>***

13. Проставим горизонтальный габаритный размер (внизу). Отключим простановку допуска в нем:

Dim: ***DIMTOL<CR>***
Размер: ***РЗМДОП<CR>***

Current value <On> New value: ***OFF<CR>***
Текущее значение <Вкл> Новое значение: ***Откл<CR>***

Проставим размер:

Dim: ***HOR<CR>***
Размер: ***ГОР<CR>***

First extension line origin or RETURN to select: ***70,110<CR>***
Начало первой выносной линии или RETURN-для выбора: ***70,110<CR>***

Second extension line origin: ***@80,0<CR>***
Начало второй выносной линии: ***@80,0<CR>***

Dimension line location: ***@0,-10<CR>***
Место положения размерной линии: ***@0,-10<CR>***

Dimension text <80>: ***80*<CR>***
Размерный текст <80>: ***80*<CR>***

14. Проставим вертикальный габаритный размер:

Dim: ***VER<CR>***
Размер: ***ВЕР<CR>***

First extension line origin or RETURN to select: ***70,110<CR>***
Начало первой выносной линии или RETURN-для выбора: ***70,110<CR>***

Second extension line origin: ***@0,120<CR>***
Начало второй выносной линии: ***@0,120<CR>***

Dimension line location: ***@-20,0<CR>***
Место положения размерной линии: ***@-20,0<CR>***

Dimension text <120>: ***120*<CR>***
Размерный текст <120>: ***120*<CR>***

15. Проставим удаление центра отверстия от нижней кромки детали (20 мм). Отменим прорисовку первой (нижней по чертежу) выносной линии, так как она уже создана при прорисовке вертикального габаритного размера:

Dim: ***DIMSE1<CR>***
Размер: ***РЗМПДВЛ1<CR>***

Current value <Off> New value: ***ON<CR>***
Текущее значение <Откл> Новое значение: ***Вкл<CR>***

Проставим размер:

Dim: ***VER<CR>***
Размер: ***ВЕР<CR>***

First extension line origin or RETURN to select: ***70,110<CR>***
Начало первой выносной линии или RETURN-для выбора: ***70,110<CR>***

Second extension line origin: ***@20,20<CR>***
Начало второй выносной линии: ***@20,20<CR>***

Dimension line location: ***60,110<CR>***
Место положения размерной линии: ***60,110<CR>***

Dimension text <20>: ***<CR>***
Размерный текст <20>: ***<CR>***

16. Проставим вертикальное межцентровое расстояние (80 мм). Включим простановку допуска:

```
Dim: DIMTOL<CR>
Размер: РЗМДОП<CR>

Current value <Off>   New value: ON<CR>
Текущее значение <Откл>  Новое значение: Вкл<CR>
```

Разрешим AutoCADу проставить размер с допуском самостоятельно:

```
Dim: CONT<CR>
Размер: ПРО<CR>
```

Введем координаты центра левого верхнего отверстия:

```
Second extension line origin: 90,210<CR>
Начало второй выносной линии: 90,210<CR>

Dimension text <80>: <CR>
Размерный текст <80>: <CR>
```

17. Сделаем выноску с указанием толщины прокладки. Заменим размерную стрелку на точку:

```
Dim: DIMBLK<CR>
Размер: РЗМБЛК<CR>

Current value <>   New value: DOT<CR>
Текущее значение <>  Новое значение: DOT<CR>
```

Установим диаметр точки равным 1 мм:

```
Dim: DIMASZ<CR>
Размер: РЗМВЛСТ<CR>

Current value <3>   New value: 1<CR>
Текущее значение <3>  Новое значение: 1<CR>
```

Заменим шрифт CYRILTLC на CYRILL:

```
Dim: STYle<CR>
Размер: ГАРнитура<CR>

New text style <CYRILTLC>: CYRILL<CR>
Новая гарнитура шрифта <CYRILTLC>: CYRILL<CR>

CYRILL is now the current text style.
CYRILL текущая гарнитура шрифта.
```

Сделаем выноску:

```
Dim: LEAder<CR>
Размер: ВЫНоска<CR>

Leader start: 140,170<CR>
Начало выноски: 140,170<CR>

To point: @30<20<CR>
К точке: @30<20<CR>

To point: @5<0<CR>
К точке: @5<0<CR>

To point: <CR>
К точке: <CR>

Dimension text <80>:  s5*<CR>
Размерный текст <80>:  s5*<CR>
```

Покинем команду DIM (РАЗМЕР):

Dim: ***EXIT<CR>***
Размер: ***ВЫХОД<CR>***

18. Сформируем текст "*Размеры для справок". Выведем на экран весь наш формат:

Command: ***ZOOM<CR>***
Команда: ***ПОКАЖИ<CR>***

All/Center/...: ***A<CR>***
Все/Центр/...: ***В<CR>***

Введем текст:

Command: ***TEXT<CR>***
Команда: ***ТЕКСТ<CR>***

Start point or Align/Center/Fit/Middle/Right/Style: ***80,80<CR>***
Начальная точка или ВПИсанный/Центр/ВЫравненный/Середина/ВПРаво/Гарнитура: ***80,80<CR>***

Height <0>: ***5<CR>***
Высота <0>: ***5<CR>***

Rotation angle <0>: ***<CR>***
Угол поворота <0>: ***<CR>***

Text: ****Размеры для справок<CR>***
Текст: ****Размеры для справок<CR>***

При работе с англоязычной версией AutoCADa может оказаться, что при наборе информации вы не увидите ее эхо-отображения в нижней зоне экрана, однако после нажатия <CR> введенный текст появится в графической зоне.

Теперь можно удалить сетку:

Command: ***GRID<CR>***
Команда: ***СЕТКА<CR>***

Grid spacing (X) or ON/OFF/Snap/Aspect <10>: ***OFF<CR>***
Интервал сетки(X) или Вкл/Откл/Шаг/Аспект <10>: ***О<CR>***

19. О формировании основной надписи.

Бланки различных форматов можно создать средствами AutoCADa заблаговременно, сохранив их на диске под различными именами. Это будет не что иное, как библиотеки форматов. Пусть, например, бланку первого листа формата A4 было присвоено имя FA4_1. Тогда вызов его с диска можно осуществить следующим образом:

Command: ***INSERT<CR>***
Команда: ***ВСТАВЬ<CR>***

Block name (or ?): ***FA4_1<CR>***
Имя блока (или ?): ***FA4_1<CR>***

Insertion point: ***0,0<CR>***
Точка вставки: ***0,0<CR>***

На остальные запросы ответим нажатием <CR>. В результате этого наша деталь "облачится" в рамку. Далее остается лишь заполнить основную надпись бланка так, как это мы делали в предыдущем шаге, и чертеж готов.

Однако такой способ формирования основной надписи достаточно утомителен, а хранение всех используемых на практике бланков форматов

займет очень много места на диске. О более эффективном способе формирования основной надписи мы упомянем позже.

20. Сохранение полученного чертежа на винчестере:

Command: ***SAVE<CR>***
Команда: ***СОХРАНИ<CR>***

File name <PROKLAD>: ***<CR>***
Имя файла <PROKLAD>: ***<CR>***

Если же потребуется сохранить чертеж на дискете, то на запрос системы об имени файла введите:

File name <PROKLAD>: ***A: \PROKLAD<CR>***
Имя файла <PROKLAD>: ***A: \PROKLAD<CR>***

21. Прорисовка чертежа на плоттере.

Прорисовать чертеж на плоттере можно либо из главного меню (рис. 2), задав режим 3, либо сразу по завершении чертежа, находясь еще в графическом редакторе (как в нашем случае):

Command: ***PLOT<CR>***
Команда: ***ЧЕРТИ<CR>***

What to plot - Display, Extents, Limits, View or Window <D>: ***<CR>***
Что чертить - Экран, Границы, Лимиты, Вид или Рамку <Э>: ***<CR>***

Далее AutoCAD ознакомит нас с протоколом настройки:

Plot will NOT be written to a selected file
Sizes are in Millimeters
Plot origin is at (0.00,0.00)
Plotting area is 863.60 wide by 546.10 high (MAX size)
Plot is NOT rotated 90 degrees
Pen width is 0.25
Area fill will NOT be adjusted for pen width
Hidden lines will NOT be removed
Scale is 1=1
Do you want to change anything? <N>: ***<CR>***

Чертеж НЕ будет записан в отдельный файл
Размеры в мм
Точка начала отсчета на чертеже (0.00,0.00)
Область черчения. Ширина-863.60; Высота-546.10 (размер МАКС)
Чертеж НЕ поворачивается на 90 градусов
Толщина пера - 0.25
Закрашивание будет производится с учетом толщины пера
Скрытые линии НЕ будут удалены
Масштаб 1=1
Хотите что-либо изменить? <Н>: ***<CR>***

Согласимся с содержимым протокола, нажав <CR>. После этого AutoCAD напомнит нам о подготовке плоттера к работе:

Effective plotting area: 476.46 wide by 297.00 high
Position paper in plotter.
Press RETURN to continue or S to Stop for hardware setup: ***<CR>***

Область чертежа: ширина - 476.46; высота - 297.00
Установите бумагу на плоттер.
Нажмите RETURN, чтобы продолжить, или О, чтобы прекратить настройку: ***<CR>***

После нажатия на <CR> начнется прорисовка чертежа на плоттере, по завершении которой на экране будет сформировано сообщение:

Plot complete.
Press RETURN to continue: ***<CR>***
Вывод на плоттер завершен.
Для продолжения нажмите RETURN: ***<CR>***

При нажатии <CR> AutoCAD передаст управление графическому редактору.

22. Выход из AutoCADa:

Command: ***QUIT<CR>***
Команда: ***ПОКИНЬ<CR>***

Really want to discard all changes to drawing? ***Y<CR>***
Действительно не нужны все изменения в рисунке? ***Д<CR>***

После этого мы попадем на главное меню AutoCADa (рис. 2). Для выхода из AutoCADa введите нулевой пункт меню:

Enter selection: ***0<CR>***
Ваш выбор: ***0<CR>***

Можно ли быстрее?

Если оценить трудоемкость только что изготовленного с помощью AutoCADa чертежа, то скептики скажут - вручную такой чертеж можно изготовить быстрее, да и обойдется он дешевле. И в чем-то они будут правы, хотя, если за пультом ЭВМ будет сидеть опытный оператор, с первым заявлением скептика можно и не согласиться. Тем не менее выпуск чертежей таким способом действительно дороговат.

Эффективность AutoCADa значительно возрастет, если, во-первых, с его помощью будут выпускаться чертежи изделий, которые по своим конструктивным признакам можно отнести к одному классу, и, во-вторых, если эти изделия будут содержать те или иные однотипные элементы. Тогда, однажды потрудившись над созданием библиотек всех таких элементов, в дальнейшем можно быстро формировать чертежи изделий данного класса. Поясним сказанное на примере.

Представим, что мы ведем разработку приборных панелей, на которых в различных сочетаниях размещаются типовые детали: тумблеры, кнопки, разъемы и другие навесные элементы. В этом случае вначале средствами AutoCADa следует создать библиотеки стандартных элементов, т.е. сформировать (так, как мы формировали чертеж прокладки) графические образы всех применяемых типов тумблеров, кнопок, разъемов и сохранить их на машинных носителях информации с присвоением каждому элементу своего имени. Правда, библиотечные элементы целесообразнее сохранять на диске не с помощью команды SAVE (СОХРАНИ), как мы делали ранее, а командой WBLOCK (ПБЛОК), в этом случае графическая информация размещается на диске компактнее:

Command: ***WBLOCK<CR>***
Команда: ***ПБЛОК<CR>***

File name: ***NAME<CR>***
Имя файла: ***NAME<CR>***

Block name: ****<CR>***
Имя блока: ****<CR>***

После завершения отработки команды WBLOCK (ПБЛОК) графический образ элемента с базовой (опорной) точкой вставки (0,0) будет записан на диск. Его полное имя - NAME.DWG. Если потребуется сместить базовую точку в другое место, отличное от (0,0), то на запрос "Block name:" или "Имя блока:" следует сразу нажать <CR> (без ввода "*"), тогда AutoCAD сделает запрос координат базовой точки вставки.

Вас пугает трудоемкость процесса создания библиотек? Но и здесь

AutoCAD может прийти вам на помощь. Представим, что мы хотим создать библиотеку прокладок (просто потому, что мы их уже умеем рисовать на ЭВМ). Выпишем в одну колонку всю информацию, которую мы вводили в ЭВМ при создании прокладки, исключив для упрощения процедуру простановки осевых линий и образмеривания детали:

```
LIMITS  210,297
ZOOM A
SNAP 10
GRID 10
PLINE 70,110 W 0.6  @120<90 @80<0 @120<270 C
ZOOM W 40,90 210,250
PLINE 100,130 A CE 90,130 A 180 CL
ARRAY L  R 2 2 80 40
```

Обладатели русскоязычной версии AutoCADа без труда сделают такую выписку по своей системе самостоятельно, если будут придерживаться следующих правил:

- в каждой строке приводится полная информация по одной команде;
- ответы на дополнительные запросы любой команды отделяются одним пробелом относительно друг друга и относительно самой команды;
- при наличии ответа по умолчанию вписывается пробел.

В результате мы получим своеобразный сценарий, по которому будет создаваться чертеж прокладки. Если теперь этот сценарий набрать с помощью любого текстового редактора (например, ЛЕКСИКОНа) и записать в виде файла на диск под произвольным именем, но с обязательным расширением ".SCR" (например, PROKL_1.SCR), то можно заставить AutoCAD "проиграть" его в автоматическом (пакетном) режиме и мгновенно получить чертеж прокладки. Редактируя с помощью ЛЕКСИКОНа сценарий (а компоненты одной библиотеки будут изменяться незначительно) и вновь запуская AutoCAD для его исполнения, мы достаточно легко создадим всю библиотеку.

Как же запустить сценарий для исполнения? После того как вы вошли в AutoCAD, задали режим 1 из главного меню, имя очередного создаваемого библиотечного элемента (PROKL_1) и оказались в графическом режиме, следует подать команду SCRIPT (ПАКЕТ):

Command: ***SCRIPT<CR>***
Команда: ***ПАКЕТ<CR>***

Script file <PROKL_1>: ***<CR>***
Пакетный файл <PROKL_1>: ***<CR>***

Появившийся в результате этого графический образ можно рассматривать и как готовый библиотечный элемент, и как заготовку, которую еще следует дорабатывать "вручную" средствами AutoCADа (все зависит от степени полноты сценария). С помощью команды WBLOCK (ПБЛОК), как это уже говорилось ранее, завершенный библиотечный элемент отправляется на сохранение.

Здесь, как вы заметили, имя сценария совпадает с именем библиотечного элемента, для которого создавался сценарий. Различие лишь в расширениях имен файлов: у имени сценария расширение ".SCR", а у имени библиотечного элемента - ".DWR". Однако это делается всего лишь для удобства разработки библиотеки и при необходимости от этого правила можно отступить.

Конечно, на создание библиотек типовых графических образов уйдет много времени, но потом этот труд вполне окупится: при формировании очередного чертежа приборной панели вам достаточно лишь вызывать по имени нужный элемент и устанавливать его в требуемое место панели:

```
Command: INSERT<CR>
Команда: ВСТАВЬ<CR>

Block name (or ?): NAME<CR>
Имя блока (или ?): NAME<CR>

Insertion point: X,Y<CR>
Точка вставки: X,Y<CR>
```

На остальные запросы следует отвечать нажатием <CR>.

Появившийся на экране монитора библиотечный элемент можно как угодно обрабатывать средствами AutoCADa, например, при необходимости размножить его по полю чертежа с помощью команды ARRAY (МАССИВ) так, как мы размножали в свое время отверстие на прокладке и т.д.

Согласитесь, формирование чертежей с использованием хорошо развитых библиотек позволит значительно ускорить процесс разработки чертежа.

Еще быстрее и удобнее

AutoCAD является универсальной графической системой. Однако это, казалось бы замечательное, свойство и не дает многим хорошим универсальным системам достичь высокой эффективности, когда дело касается частного случая (именно универсальность и делает их неуклюжими, обремененными ненужными функциями).

К большому преимуществу AutoCADa следует отнести тот факт, что его достаточно легко можно адаптировать к выпуску специальной продукции, т.е. AutoCAD на время становится узкоспециализированным. И самое главное, такую адаптацию способен осуществить сам пользователь, не прибегая к услугам разработчика системы. Достигается это с помощью специального языка AutoLISP [2], встроенного в систему и понятного ей. К основным достоинствам этого языка следует отнести его гибкость, простоту использования.

AutoLISP по своему синтаксису наиболее близок к языку программирования CommonLISP [3], широко используемому в задачах искусственного интеллекта. И тот и другой основаны на вычислении функций, которые записываются в префиксной или польской форме, т.е. вначале следует запись операции (действия), а затем перечисляются элементы, участвующие в операции (операнды). Вся конструкция заключается в скобки. В качестве примера можно привести функцию вычитания четырех чисел:

(- 10 2 3 1)

Результатом вычисления этой функции будет число 4.

Разработчики AutoLISPa ввели в этот язык конструкции, позволяющие использовать возможности AutoCADa.

Теперь рассмотрим, как специализировать AutoCAD, например, на разработку чертежей известной уже нам прокладки (предположим, что нам то и дело приходится выпускать чертежи таких прокладок с неизменной формой, но самых разнообразных размеров).

Вновь распишем наши действия по шагам.

1. Сформулируем свои требования к AutoCADу, т.е. опишем, каким бы мы хотели видеть его в работе, его алгоритм "поведения" при проектировании прокладок, укажем некоторые фиксированные исходные данные:

А) После запуска из AutoCADa разработанной нами прикладной программы проектирования прокладок на экране монитора должен появиться слайд (рисунок) прокладки с указанием обозначений изменяющихся размеров (рис. 4).

Б) AutoCAD последовательно запрашивает у оператора информацию о всех размерах прокладки в форме:

"Введите размер LGAB:"
"Введите размер L1:" и т.д.

Последним запрашивается не указанный на слайде параметр:

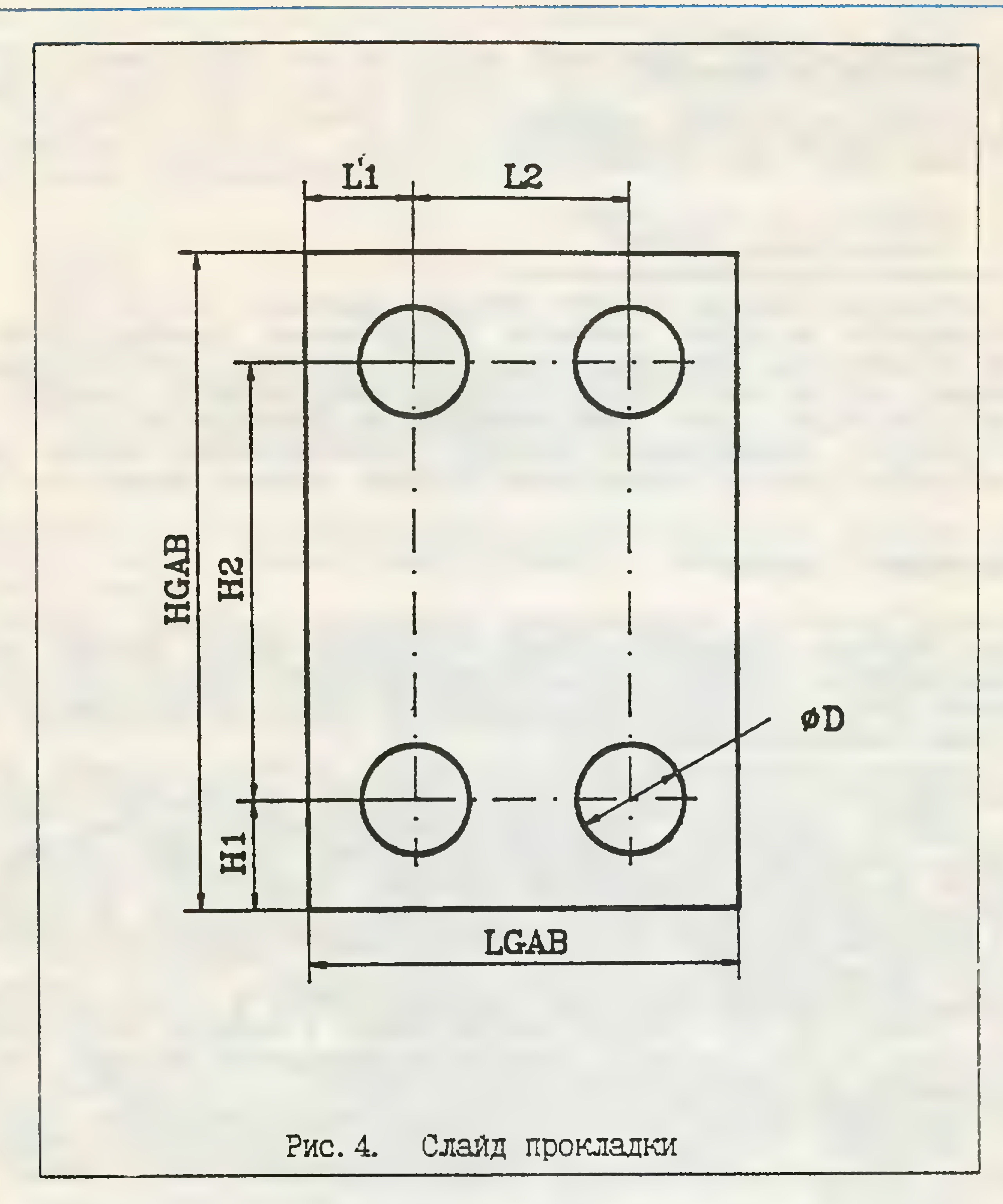


Рис. 4. Слайд прокладки

"Введите толщину прокладки:"
Для упрощения программы введем такие ограничения: вводимые размеры могут быть любыми целыми числами, размер допуска на L2 и H2 фиксирован и равен 1 мм в обе стороны, размеры LGAB, HGAB и толщина прокладки - справочные.

В) По завершении ввода последнего параметра AutoCAD в автоматическом режиме должен сформировать чертеж прокладки в соответствии с введенными размерами.

Г) Оператор "облачает" чертеж прокладки в подходящий формат, вписывает средствами AutoCADа фразу "*Размеры для справок", заполняет основную надпись, и чертеж готов.

2. Подготовим слайд с видом прокладки.

Войдите в AutoCAD и в режиме 1 зарегистрируйте будущий рисунок, например, под именем RIS_4. Повторите формирование чертежа прокладки вплоть до прорисовки выноски с указанием толщины изделия (на слайде выноску делать не будем). Параметры прокладки возьмите прежние, однако вместо шрифтов CYRILL и CYRILTLC установите шрифт ROMANC - он содержит одновременно латинский шрифт и знак диаметра (последнему соответствует буква "Б"), кроме того, на запросы "Dimension text <...>:" ("Размерный текст <...>:") отвечайте вводом обозначений размеров (см. рис. 4). После этого подайте команду MSLIDE (ДСЛАЙД):

```
Command: MSLIDE<CR>
Команда: ДСЛАЙД<CR>

Slide file <RIS_4>: <CR>
Слайд файл <RIS_4>: <CR>
```

В результате рисунок, изображенный на экране монитора, будет оформлен в виде слайда и записан на диск под именем RIS_4.SLD.

Покиньте AutoCAD.

3. Ниже приводится текст программы на AutoLISPe, с помощью которой в диалоговом режиме будут формироваться чертежи прокладок. Программа составлена для англоязычной версии AutoCADa, чтобы приспособить ее к русскоязычной версии, следует лишь скорректировать названия команд для AutoCADa и дополнительные ответы на них (в исходном тексте программы заменяемый текст выделен жирным шрифтом). Например, строка (command **"VSLIDE"** "RIS_4") заменяется на (command "СЛАЙД" "RIS _4") и т.д.

Текст программы снабжен подробными комментариями. К комментариям в AutoLISPe относится любой текст, находящийся справа от знака точки с запятой и продолжающийся до конца строки. AutoLISP игнорирует для себя все комментарии, встречающиеся в тексте прграммы, но оставляет их для пользователей. Итак, программа:

```
;ЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖ
;Ж   П Р О К Л А Д К А   Ж
;Ж   (ЛИСП-программа)     Ж
;ЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖЖ

;Для нормальной работы настоящей программы требуется иметь на
;винчестере файл RIS_4.SLD, представляющий слайд с изображе-
;нием общего вида прокладки.


;*************************************************
;*   ОПИСАНИЕ БЕЗАРГУМЕНТНОЙ ФУНКЦИИ SLIDE,      *
;*   ВЫЗЫВАЮЩЕЙ СЛАЙД RIS_4 НА ЭКРАН МОНИТОРА    *
;*************************************************
(defun SLIDE ()
    (command "VSLIDE" "RIS_4")
);завершение определителя функций "defun"


;*************************************************
;*   ОПИСАНИЕ БЕЗАРГУМЕНТНОЙ ФУНКЦИИ ENTER,      *
;*   ОСУЩЕСТВЛЯЮЩЕЙ ВВОД ИСХОДНЫХ ДАННЫХ         *
;*************************************************
(defun ENTER ()
   ;Организация диалога с пользователем и присвоение переменным LGAB,
   ;L1,...,S значений, введенных пользователем:
   (setq LGAB (getdist "\nВведите размер LGAB:"))
   (setq L1 (getdist "\nВведите размер L1:"))
   (setq L2 (getdist "\nВведите размер L2:"))
   (setq HGAB (getdist "\nВведите размер HGAB:"))
   (setq H1 (getdist "\nВведите размер H1:"))
   (setq H2 (getdist "\nВведите размер H2:"))
   (setq D (getdist "\nВведите размер D:"))
   (setq S (getdist "\nВведите толщину прокладки:"))
);завершение определителя функций "defun"
```

```
;***********************************************
;*   ОПИСАНИЕ БЕЗАРГУМЕНТНОЙ  ФУНКЦИИ  VID,      *
;*   ОСУЩЕСТВЛЯЮЩЕЙ ПРОРИСОВКУ ПРОКЛАДКИ         *
;*       (БЕЗ ПРОСТАНОВКИ РАЗМЕРОВ)              *
;***********************************************
(defun VID ()
   (command "REDRAW") ;удаление слайда с экрана
   (command "LIMITS" "" (list (+ LGAB 80) (+ HGAB 40)))
   ;Здесь координаты левого нижнего угла берутся по умолчании  (приз-
   ;нак - ""), а верхнего правого угла вычисляются: координата  X  на
   ;80 мм больше LGAB (чтобы поместились размеры и фраза "4 отв..."),
   ;координата Y на 40 мм больше HGAB (чтобы поместились размеры).
   (command "ZOOM" "A")
   (command "PLINE"                        ;прорисовка контура прокладки
       "30,20"                             ;от точки с координатами {30,20}
                                           ;  (левый нижний угол прокладки)
       "W" "0.6" ""                        ;задание ширины линии (0.6 мм)
       (list 30 (+ 20 HGAB))               ;задание координат  верхней  ле-
                                           ;              вой точки прокладки
       (list (+ 30 LGAB) (+ 20 HGAB))      ;задание координат  верхней пра-
                                           ;              вой точки прокладки
       (list (+ 30 LGAB) 20)               ;задание координат  нижней  пра-
                                           ;              вой точки прокладки
       "C"                                 ;контур замкнуть
   ;--- повторный вызов той же команды для изображения отверстия: ---
           "PLINE"
       (list (+ 30 L1 (/ D 2)) (+ 20 H1));от крайней правой точки от-
                                         ;                        верстия
       "A" "CE"
       (list (+ 30 L1) (+ 20 H1))        ;координаты центра отверстия
       "A" "180"                         ;чертить дугу в 180 градусов
       "CL"                              ;замкнуть дугу по кругу
   ;--- размножение окружности до четырех: ---
           "ARRAY"
       "L" "" "R" "2" "2" H2 L2
   );завершение функции "command"
);завершение определителя функций "defun"


;************************************************
;*   ОПИСАНИЕ БЕЗАРГУМЕНТНОЙ  ФУНКЦИИ  STRIH,     *
;*      ИЗОБРАЖАЮЩЕЙ ШТРИХ-ПУНКТИРНЫЕ ЛИНИИ       *
;************************************************
(defun STRIH ()
   ;--- загрузка и установка штрих-пунктирной линии: ---
   (command "LINETYPE"
       "L" "DASHDOT" "" "" "S" "DASHDOT"
       ""                                ;завершение команды
   ;--- изображение верхней горизонтальной линии: ---
           "LINE"
       (list (+ 30 L1) (+ 20 H1 H2))     ;координаты центра  верхнего
                                         ;           левого отверстия
       (list (+ 30 L1 L2 (/ D 2) 2) (+ 20 H1 H2)) ;координаты правого
                                        ;  конца штрих-пунктирной линии
       ""                               ;завершение команды
   ;--- изображение нижней горизонтальной линии: ---
           "LINE"
       (list (+ 30 L1) (+ 20 H1))
       (list (+ 30 L1 L2 (/ D 2) 2) (+ 20 H1))
       ""
   ;--- изображение левой вертикальной линии: ---
           "LINE"
```

```
        (list (+ 30 L1) (+ 20 H1 H2))
        (list (+ 30 L1) (- (+ 20 H1) (/ D 2) 2))
        ""
   ;--- изображение правой вертикальной линии: ---
            "LINE"
        (list (+ 30 L1 L2) (+ 20 H1 H2))
        (list (+ 30 L1 L2) (- (+ 20 H1) (/ D 2) 2))
        ""
   ;--- восстановление сплошной основной линии: ---
            "LINETYPE" "S" "CONTINUOUS" ""
   );завершение функции "command"
);завершение определителя функции "defun"


;*************************************************
;*   ОПИСАНИЕ БЕЗАРГУМЕНТНОЙ  ФУНКЦИИ  RAZM,     *
;*   ОСУЩЕСТВЛЯЮЩЕЙ ОБРАЗМЕРИВАНИЕ ПРОКЛАДКИ     *
;*************************************************
(defun RAZM ()
   (setq a (fix D)) ;присвоить переменной "a" значение переменной "D"
   (setq b 0)       ;сброс счетчика пробелов
   (while (> a 21)  ;пока величина "a" превышает длину фразы "4 отв Ø"
                    ;       (ширина одного символа, в том числе и про-
                    ;        бела, - 3 мм), производить действия:
             (setq a (- a 3))
             (setq b (+ b 1))
         ;здесь  "b" - минимальное количество пробелов,  необходимых
         ; для формирования AutoCADом сообщения "Text does not fit."
         ;   ("Текст не помещается.") и запроса о параметрах выноски
   );завершение функции "while"
   (setq c " ")
   (repeat b (setq c (strcat c " ")))
         ;здесь "c" - строка из "b" пробелов
   ;--- установка формата единицы измерения: ---
   (command "UNITS" "" "0" "" "" "" ""
   ;--- установка переменных параметров размерных линий: ---
            "DIM"
       "DIMTM" "1" "DIMTP" "1" "DIMTXT" "4"
       "EXIT"                            ;завершение команды
   ;--- настройка шрифта CYRILL: ---
            "STYLE"
       "CYRILL" "CYRILL" "" "" "" "" "" ""
   ;--- настройка шрифта CYRILTLC: ---
            "STYLE"
       "CYRILTLC" "CYRILTLC" "4" "" "" "" "" ""
   ;--- простановка размера на отверстии: ---
            "DIM"
       "DIAM" (list (+ 30 L1 L2 (* (sqrt 2) (/ D 4)))
                    (+ 20 H1 (* (sqrt 2) (/ D 4))))
       (strcat "4 otv Б" (itoa (fix D)) c)     ;монтаж строки из трех
                                               ;             фрагментов
       (list (+ 30 LGAB 5) (+ 20 H1 (- LGAB L1 L2) 5))
   ;--- простановка размеров L1 и L2: ---
       "HOR" (list 30 (+ 20 HGAB))
             (list (+ 30 L1) (+ 20 H1 H2))
             (list 30 (+ 20 HGAB 10)) ""
       "DIMTOL" "ON" ;включение простановки допуска
       "CONT" (list (+ 30 L1 L2) (+ 20 H1 H2)) ""
   ;--- простановка размера LGAB: ---
       "DIMTOL" "OFF" ;отключение простановки допуска
       "HOR" (list 30 20)
             (list (+ 30 LGAB) 20)
```

```
            (list 30 10) (strcat (itoa (fix LGAB)) "*")
   ;--- простановка размера HGAB: ---
       "VER" (list 30 20)
             (list 30 (+ 20 HGAB))
             (list 10 20) (strcat (itoa (fix HGAB)) "*")
   ;--- простановка размеров H1 и H2: ---
       "DIMSE1" "ON" ;отмена прорисовки нижней выносной линии
       "VER" (list 30 20)
             (list (+ 30 L1) (+ 20 H1))
             (list 20 20) ""
       "DIMTOL" "ON" ;включение простановки допуска
       "CONT" (list (+ 30 L1) (+ 20 H1 H2)) ""
   ;--- указание толщины прокладки (с помощью выноски): ---
       "DIMBLK" "DOT";замена размерной стрелки на точку
       "DIMASZ" "1"  ;установка диаметра точки равным 1 мм
       "STY" "CYRILL";установка нового шрифта
       "LEA" (list (+ 30 L1 L2 (/ (- LGAB L1 L2) 2))
                   (+ 20 H1 (/ H2 2)))
             (list (+ 30 LGAB 5)
                   (+ 20 H1 (/ H2 2) (+ 5 (/ (- LGAB L1 L2) 2))))
             (list (+ 30 LGAB 10)
                   (+ 20 H1 (/ H2 2) (+ 5 (/ (- LGAB L1 L2) 2))))
             "" (strcat " s" (itoa (fix S)) "*")
       "EXIT"
       "REDRAW"
   );завершение функции "command"
);завершение определителя функции "defun"


;*****************************************************************
;*  КОРНЕВАЯ БЕЗАРГУМЕНТНАЯ ПРОГРАММА PROKLAD, СОБИРАЮЩАЯ   *
;*  ФУНКЦИИ SLIDE, ENTER, VID, STRIH, RAZM ВОЕДИНО          *
;*****************************************************************
   (defun C:PROKLAD ()              ;для русскоязычной версии  устано-
                                    ;   вите имя корневой  программы -
                                    ;                          ПРОКЛАД
   (setq BLIP (getvar "BLIPMODE"))  ;сохранение  системной  переменной
                                    ; BLIPMODE, управляющей простанов-
                                    ;   кой засечек на экране монитора
   (setq CMD (getvar "CMDECHO"))    ;сохранение  системной  переменной
                                    ; CMDECHO, управляющей выводом  на
                                    ;  экран командного эха
   (setvar "BLIPMODE" 0)            ;отключение простановки засечек
   (setvar "CMDECHO" 0)             ;отключение вывода командного эха
   (SLIDE)                          ;вызов функции SLIDE
   (ENTER)                          ;вызов функции ENTER
   (VID)                            ;вызов функции VID
   (STRIH)                          ;вызов функции STRIH
   (RAZM)                           ;вызов функции RAZM
   (setvar "BLIPMODE" BLIP)         ;восстановление прежнего  значения
                                    ;   системной переменной BLIPMODE
   (setvar "CMDECHO" CMD)           ;восстановление прежнего  значения
                                    ;   системной переменной  CMDECHO
);завершение определителя функции "defun"
```

4. Наберите текст программы с помощью любого текстового редактора и запишите на диск в виде файла с именем PROKLAD.LSP.

5. С помощью нашей программы создадим прокладку с произвольными размерами. Для этого войдите в AutoCAD и в режиме 1 введите имя будущего чертежа (какое хотите). Загрузите программу PROKLAD.LSP:

Command: ***(LOAD "PROKLAD")<CR>***
Команда: ***(LOAD "PROKLAD")<CR>***

и запустите ее в работу:

Command: ***PROKLAD<CR>***
Команда: ***ПРОКЛАД<CR>***

На экране монитора появится слайд прокладки и последуют запросы о ее параметрах. По окончании ввода последнего параметра слайд исчезнет и начнется автоматическое вычерчивание прокладки с заданными размерами. Далее выберите подходящий формат и, задав его с помощью команды LIMITS (ЛИМИТЫ), перенесите прокладку в центр этого формата:

Command: ***MOVE<CR>***
Команда: ***ПЕРЕНЕСИ<CR>***

Сообщите AutoCADу об объекте переноса, заключив всю прокладку вместе с нанесенными на нее размерами в воображаемую рамку:

Select objects: ***W<CR>***
Выберите объекты: ***P<CR>***

First corner: ***0,0<CR>***
Первый угол: ***0,0<CR>***

Далее введите численные координаты верхнего правого угла воображаемой рамки с учетом реальных значений LGAB и HGAB:

Other corner: ***LGAB+80,HGAB+40<CR>***
Другой угол: ***LGAB+80,HGAB+40<CR>***

20 found.
Select objects: ***<CR>***
20 найден(ы).
Выберите объекты: ***<CR>***

В качестве базовой точки выберем левый нижний угол прокладки:

Base point or displacement: ***30,20<CR>***
Базовая точка или перемещение: ***30,20<CR>***

Введите численные координаты той точки перемещения, куда будет помещена базовая точка:

Second point of displacement: ***X,Y<CR>***
Вторая точка перемещения: ***X,Y<CR>***

Остается сформировать и заполнить основную надпись и вписать фразу "*Размеры для справок". Это вы сможете сделать уже и сами.

Следует отметить, что немного усложнив LISP-описание нашей программы, можно поручить AutoCADу самостоятельно завершить формирование чертежа, вплоть до автоматического выбора нужного формата, переноса прокладки в центр этого формата, формирования текстовой строки "*Размеры для справок" и заполнения основной надписи, запросив разве что у оператора фамилии лиц, имеющих отношение к выпуску чертежа.

Разрабатываем новые команды для AutoCADa

Как вы уже заметили, созданное и загруженное в AutoCAD LISP-описание прокладки стало ничем иным, как новой командой, хотя и очень "экзотической". Действительно, запуск ее осуществляется обычным способом по запросу AutoCADa "Command: " ("Команда: ").

Единственное, что, наверное, смущает пользователя - необходимость предварительной загрузки LISP-описания командой "(load ...)". Однако и этого можно избежать, если описание будет записано на диск в специальный файл с именем ACAD.LSP (а не PROKLAD.LSP, как в нашем случае). Тогда при вызове и входе в AutoCAD система сама осуществит загрузку LISP-описания из упомянутого файла.

В файл ACAD.LSP можно записать и несколько LISP-описаний, имеющих

различные корневые программы со своими именами. Таким образом можно расширить стандартный набор команд AutoCADа новыми полезными для вас командами.

Можно, например, разработать универсальную команду, по которой будет формироваться любой формат (основная надпись) чертежа. Использование такой команды не только освободит пользователя от утомительной оформительской работы, но и позволит экономно использовать дисковое пространство. Действительно, вместо того чтобы хранить на диске полный набор всевозможных форматов, достаточно иметь там лишь бланк основной надписи и бланки дополнительных граф. Тогда, запросив у пользователя обозначение требуемого формата (А0...А4) и его ориентацию (вертикальная или горизонтальная), AutoCAD сможет самостоятельно расставить бланки в нужные места (в соответствии с заказанным форматом) и нарисовать внутреннюю и внешнюю рамки формата. После этого должны последовать запросы по заполнению основной надписи: "Наименование изделия", "Обозначение документа" и др. Одновременно с этим можно предусмотреть показ крупным планом (на экране монитора) основной надписи с подсветкой очередной запрашиваемой графы. Ответы пользователя, вводимые им с клавиатуры, должны направляться AutoCADом в соответствующие графы.

Попытайтесь самостоятельно разработать такую команду. При желании можно продолжить ее совершенствование, например организовать автоматизированную простановку знака шероховатости (если он требуется) в правом верхнем углу чертежа и др.

Не жалейте времени на создание любой новой команды: тщательно продумывайте сценарий ее работы, старайтесь "взвалить" на свою программу весь рутинный труд, сопутствующий решаемой задаче. Работа с вашей командой должна доставлять удовольствие пользователю.

Заключение

В небольшой по объему статье были показаны лишь самые скромные возможности AutoCADа. В действительности они поистине неисчерпаемы. Вы, наверное, уже догадались: возможности AutoCADа в полной мере раскрываются при использовании языка AutoLISP. Наиболее заманчиво, хотя и намного сложнее, создавать такие LISP-описания, с помощью которых будет осуществляться автоматическое проектирование (а не вычерчивание) тех или иных изделий. И языковые средства AutoLISPа позволяют решать такие задачи.

Литература

1. Бергхаузер Т., Шлив П. Система автоматизированного проектирования AutoCAD: Справочник/Пер. с англ. -М.:Радио и связь, 1989. -256 с.
2. АВТОЛИСП. Версия 10: Руководство по программированию. AUTODESK LTD, 1989.
3. Хювёнен Э., Сеппянен Й. Мир Лиспа. В 2-х т. /Пер. с финск. -М.:Мир, 1990.

Роман Полонский, МАН "Искатель"
г.Симферополь

ПРЕДСТАВЛЕНИЕ И ИЗОБРАЖЕНИЕ ПРЯМОУГОЛЬНЫХ ФИГУР С ПОМОЩЬЮ ДЕРЕВА ОТРЕЗКОВ

Прямоугольными фигурами (ПФ) назовем фигуры, составленные из отрезков, параллельных осям координат.

Некоторые классы ПФ представляют большой интерес в решении ряда прикладных задач. К ним относятся, например, задача описания и обработки комбинационных логических схем, задача размещения прямоугольных элементов (блоков) на платах электронных устройств, на планах или на печатных страницах и другие (см. рис. 1,а, 1,б, 1,в).

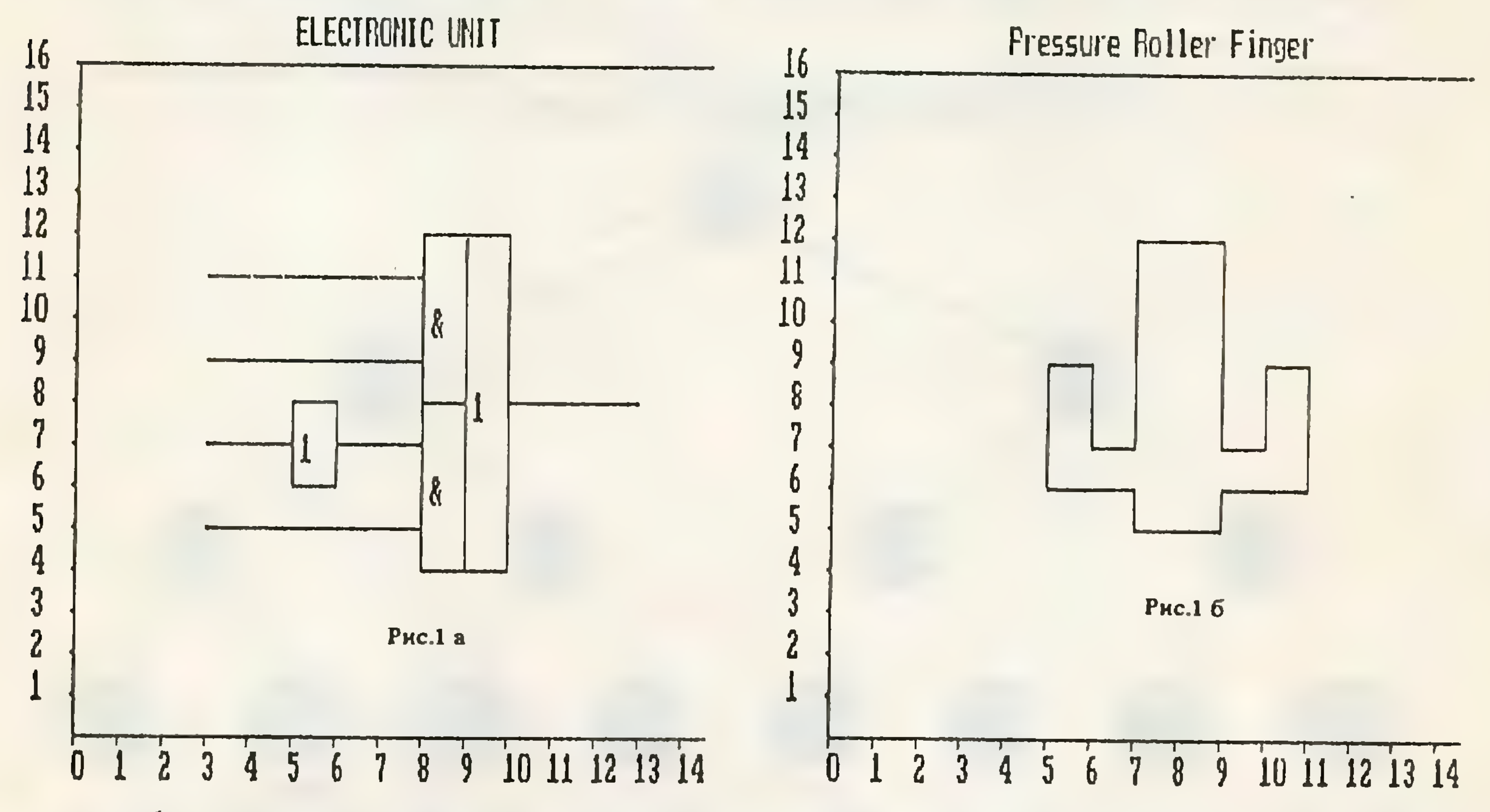


Рис.1 а

Рис.1 б

Проблемы представления ПФ возникают, если требуется строить сложные ПФ из простых, например из прямоугольников, применяя к ним теоретико-множественные операции объединения, пересечения и вычитания, если необходимо динамическое редактирование ПФ и в ряде других случаев.

Будем строить ПФ в области w = [0; w] * [0;w], т.е. в квадратном окне (рабочем поле) со стороной длины w, w натуральное.

Зададим в области w прямоугольную систему координат:

$X, Y \in \{0, 1, \dots, w\}$.

Очевидно, что всякую ПФ в области w можно задать наборами вертикальных и горизонтальных отрезков, если каждому отрезку [a;b] приписать величину Dist(a,b) — его расстояние от оси абсцисс (или ординат).

Для представления и обработки ПФ в области w будем использовать так называемое дерево отрезков, обозначаемое T(0,w).

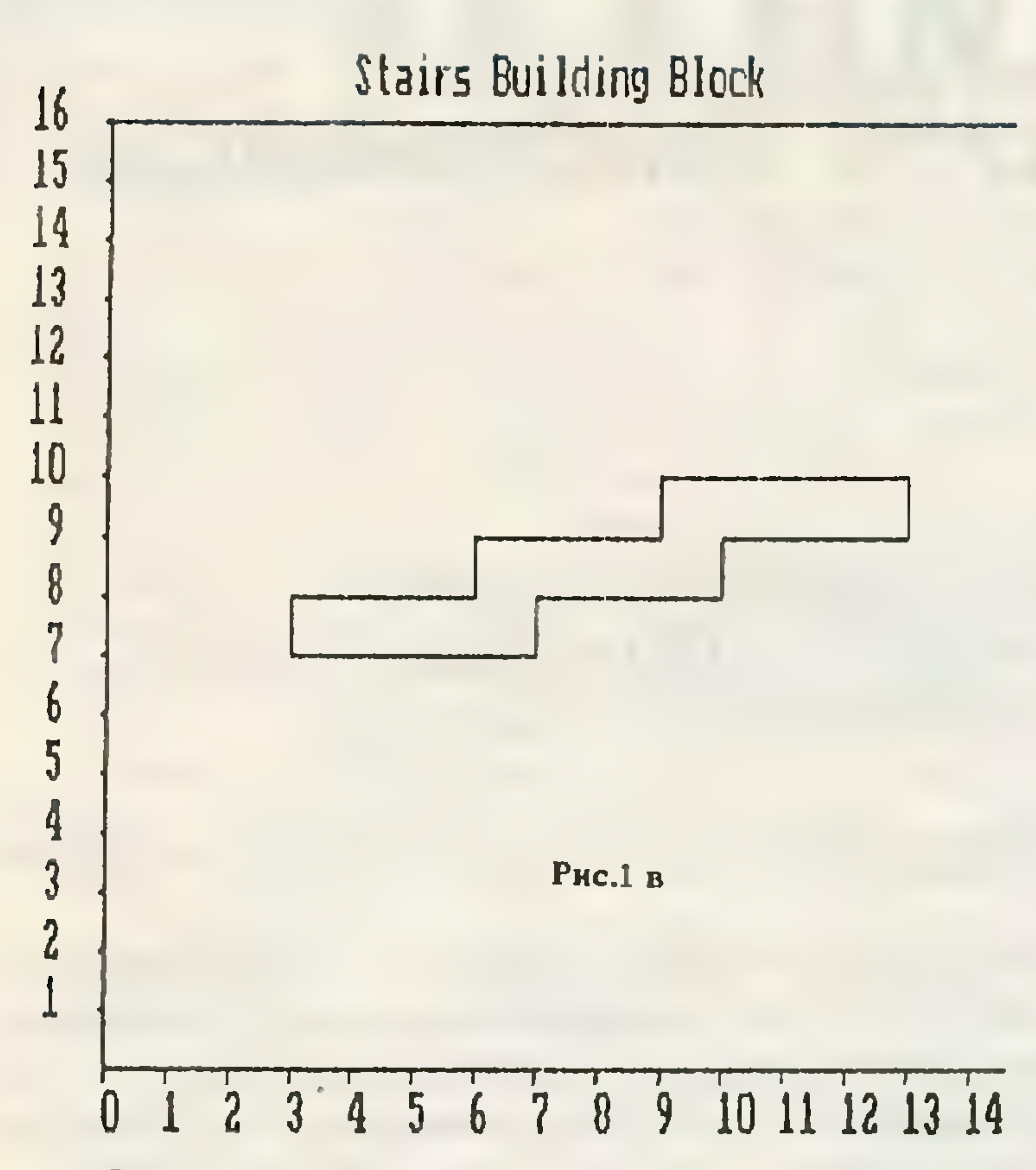


Рис.1 в

T(0,w) есть бинарное сбалансированное по высоте дерево[1], предназначенное для динамического хранения отрезков, концы которых принадлежат множеству

E = { O, 1, 2 ... , w-1, w}.

Отрезок U = [0;w] называется универсальным отрезком.

Дерево T(L,R) строится по следующему рекурсивному алгоритму.

Алгоритм построения Бинарного Дерева Отрезков (L начальное равно 0, R начальное равно w).

1. Построить корневую вершину T(L,R) и поместить в нее отрезок [L;R].

2. Если длина отрезка [L;R] > 1, то получить середину отрезка M = (L+R) div 2 и построить поддеревья дерева T(L,R): левое T(L,M) и правое T(M,R).

При L начальном, равном 0, и при R начальном, равном w=16, получим дерево, показанное на рис. 2.

Описанный выше алгоритм реализован в виде процедуры-функции ITree

THE BINARY TREE OF INTERVALS

0;16

0;8 8;16

0;4 4;8 8;12 12;16

0;2 2;4 4;6 6;8 8;10 10;12 12;14 14;16

0;1 1;2 2;3 3;4 4;5 5;6 6;7 7;8 8;9 9;10 10;11 11;12 12;13 13;14 14;15 15;16

Рис.2

1 *Бинарное дерево T сбалансировано по высоте, если его концевые вершины (листья) принадлежат двум смежным уровням T*

I. Процедура построения Дерева Отрезков

```
FUNCTION ITree (L,R: Integer): Ref;
  Var
    Mid  : Integer;
       T : Ref     ;
       s  : String  ;

  Begin
    If R<=L
          Then
             Begin
                ITree:=Nil:
                Exit;
             End;

    If
       (R-L)>=1
          Then
             Begin
                New(T);

                IF T=Nil Then Begin
                                   ClrScr     ;
                                   WriteLn ("NO NEW MEMORY");
                                   s : = ReadKey;
                                   Exit
                                End;

                T^.a          :   = l    ;
                T^.b          :   = r    ;
                T^.List_Id :   = Nil ;
                ITree         :   = t    ;

             End;

    If
       (R-L)=1
          Then
             Begin
                ITree^.LLink : = Nil;
                ITree^.RLink : = Nil;
             End
          Else
             Begin
                Mid := (R+L) div 2;
                ITree^.LLink := ITree(L,Mid);
                ITree^.RLink := ITree(Mid,R);
             End;
End;
```

Отрезки, попавшие в вершины дерева, построенного по данному алгоритму, называются **стандартными** относительно данного универсального отрезка.

Стандартных отрезков столько же, сколько вершин (узлов) бинарного дерева. Для w = 2^k стандартных отрезков 2 ^(k+1) - 1. Общее число отрезков, содержащихся в универсальном отрезке, равно w(w -1) div 2. Видно, что для достаточно больших значений w стандартные отрезки составляют лишь небольшую часть от общего числа отрезков. Например, для w = 1024 количество стандартных отрезков равно 2047, а всех отрезков — 1041552.

Итак, стандартные отрезки образуют правильное подмножество всех отрезков, содержащихся в U. Они обладают следующим важным свойством.

А именно, справедливо утверждение:

"Любой отрезок [a;b], содержащийся в U, можно представить с помощью суммы стандартных отрезков".

Задание [a;b] в виде некоторой суммы стандартных отрезков называется **фрагментацей** (в частном случае [a,b] — стандартный отрезок), а фрагментация на дереве отрезков выполняется пометкой вершин дерева, отрезки которых входят в состав [a,b].

Так как дерево отрезков (по построению) содержит все отрезки единичной длины из U, то приведенное выше утверждение легко доказывается. Желательно, однако, чобы сумма, представляющая отрезок, содержала минимальное число слагаемых фрагментов.

Рассмотрим алгоритм, который дает оптимальную фрагментацию, т.е. выбирает для представления [a;b] стандартные отрезки наибольшей длины на каждом шаге работы. Правильность алгоритма гарантируется, если [a;b] содержится в U и построено дерево T(U).

Алгоритм фрагментации отрезка [a;b] в дереве T [L, R) реализован в виде приведенной ниже процедуры Tick_Int.

II.Процедура фрагментации

```
PROCEDURE Tick_Int(T: Ref; L,R, Dist: Integer);

Var s     : String;
    P     : Id;
    Mid : Integer;

Begin
 If T <> Nil
   Then
     Begin
       If
         (L <= T^.A)
           and
         (T^.B <= R)

           Then

             Begin

               New(P);

               IF P=Nil Then
                     Begin
                       ClrScr;
                       WriteLn("NO NEW MEMORY");
                       s := ReadKey;
                       Exit
                     End;

               P^.Ref     := T^.List_Id;
               P^.Dist    := Dist;
               T^.List_Id := P;

             End

           Else
```

```
                    Begin
                        Mid := (T^.A + T^.B) div 2;
                        If L < Mid Then Tick_Int (T^.LLink,L,R,Dist);
                        If Mid < R Then Tick_Int (T^.RLink,L,R,Dist);
                    End;

          End;
     End;
```

В процедуре Tick_Int величины T^.a и T^.b задают стандартный отрезок вершины дерева, на которую указывает T, T^.LLink и T^.RLink указатели соответственно на левое и правое поддеревья вершины, на которую указывает T.

Итак, дерево T(U) — T(L,R) обеспечивает представление всех отрезков универсального отрезка U в виде суммы минимального числа стандартных отрезков (фрагментов).

Применим дерево отрезков для хранения и изображения прямоугольных фигур. Будем вносить в дерево отрезков как горизонтальные, так и вертикальные отрезки ПФ. Фрагменты вносимых отрезков будем помечать, приписывая соответствующим вершинам дерева расстояние стандартного отрезка от оси OX или OY.

Так как мы разместили окно w в 1 квадранте системы координат, то расстояния отрезков от осей всегда положительны. Воспользуемся этим обстоятельством для указания того, каким является отрезок — вертикальным или горизонтальным.

Вертикальным отрезкам, находящимся на расстоянии d от оси OX, будем приписывать величину d, горизонтальным, находящимся на расстоянии d от оси OY, — величину -d.

Один и тот же стандартный отрезок может входить в разные отрезки представляемой ПФ, поэтому вершине дерева приписывается список расстояний.

Рассмотрим в деталях процесс построения ПФ, показанной на рис.1,а, в окне W = U*U, U=[0;16], с помощью Паскаль-программы Bynary_Tree_of_Intervals.

Ниже приводится начало этой программы, в которой задаются необходимые ссылочные типы и записи, используемые для представления дерева отрезков и линейных списков расстояний, приписываемых вершинам этого дерева.

```
     Program Bynary_Tree_of_Intervals;

               ┌──────────────────────────────────────┐
               │ DRAWING A FIGURE USING THE BYNARY    │
               │        TREE OF INTERVALS             │
               │           P. i. a. 1 9 9 1           │
               └──────────────────────────────────────┘

     Uses Crt, Printer, Graph;
        ......................................

     TYPE

        Ref = ^Tree_Elem ;
        Id  = ^List_Elem  ;
        List_Elem = RECORD

             Dist  :  Integer;
             Ref   :  Id      ;

                  END;

        Tree_Elem = RECORD

             Llink,
             Rlink   :   Ref     ;
```

```
A,B     :  Integer ;
List_Id :  Id      ;

      END;
```

....

На первом этапе работы программы с помощью процедуры ITree строится бинарное дерево, изображенное на рис.2.

Затем осуществляется ввод координат концов отрезков ПФ (протокол ввода см. в табл. 1).

Таблица I

```
      I  N  P  U  T

Enter The Number of Intervals...  16

 1.     (X1:Y1) ...   3 11   (X2:Y2) ...   8 11
 2.     (X1:Y1) ...   3  9   (X2:Y2) ...   8  9
 3.     (X1:Y1) ...   3  7   (X2:Y2) ...   5  7
 4.     (X1:Y1) ...   6  7   (X2:Y2) ...   8  7
 5.     (X1:Y1) ...   3  5   (X2:Y2) ...   8  5
 6.     (X1:Y1) ...  10  8   (X2:Y2) ...  13  8
 7.     (X1:Y1) ...   5  6   (X2:Y2) ...   6  6
 8.     (X1:Y1) ...   5  9   (X2:Y2) ...   6  8
 9.     (X1:Y1) ...   5  6   (X2:Y2) ...   5  8
10.     (X1:Y1) ...   6  6   (X2:Y2) ...   6  8
11.     (X1:Y1) ...   8  4   (X2:Y2) ...  10  4
12.     (X1:Y1) ...   8 12   (X2:Y2) ...  10 12
13.     (X1:Y1) ...   8  4   (X2:Y2) ...   8 12
14.     (X1:Y1) ...  10  4   (X2:Y2) ...  10 12
15.     (X1:Y1) ...   9  4   (X2:Y2) ...   9 12
16.     (X1:Y1) ...   8  8   (X2:Y2) ...   9  8
```

Таблица 2

```
The Ticked Nodes List
Standard Interval [ 3: 4].  Distances:   -5  -7  -9 -11
Standard Interval [ 4: 8].  Distances:    9  10   9  -5  -9 -11
Standard Interval [ 4: 5].  Distances:   -7
Standard Interval [ 5: 6].  Distances:   -8  -6
Standard Interval [ 6: 8].  Distances:    6   5  -7
Standard Interval [ 8:12].  Distances:    9  10   8
Standard Interval [ 8:10].  Distances:  -12  -4
Standard Interval [ 9: [illegible]].  Distances:   -9
Standard Interval [10:12].  Distances:   -8
Standard Interval [12:13].  Distances:   -8
```

После ввода координат концов очередного, i-го, отрезка вычисляется расстояние d этого отрезка от параллельной ему оси и определяется проекция [a;b] введенного отрезка на эту ось.

Проекция [a;b] и расстояние d отрезка от оси (d>0 для вертикального и d<0 для горизонтального) подаются на вход процедуры Tick_Int (T,a,b,d), которая при каждом обращении приписывает стандартным отрезкам — фрагментам отрезка [a;b] величину d.

В табл. 2 показан протокол фрагментации отрезков ПФ стандартными отрезками. Например, стандартный отрезок [6;8] входит в ПФ трижды: два раза как вертикальный на расстояниях 6 и 5 от оси OY и один раз как горизонтальный на расстоянии 7 от оси OX.

После завершения этапа фрагментации ПФ представлена в памяти в виде комбинированной структуры — бинарного дерева с присоединенными к некоторым его вершинам списками расстояний. Эта структура содержит всю информацию, необходимую для изображения (изготовления) ПФ.

Повторим, что структура является динамической, позволяющей редактировать (изменять) ПФ.

Изображение ПФ на экране выполняется с помощью процедуры Scan_n_Draw (T), которая просматривает дерево отрезков, обнаруживает помеченные стандартные отрезки, принадлежащие ПФ, и рисует их на экране дисплея.

III. Процедура изображения ПФ.

```
PROCEDURE Scan_n_Draw ( T: Ref );

Var s         : String ;
    a,b,Dist  : Integer;
    P         : Id      ;
```

```
Begin
    If T<>Nil
      Then
        Begin
          If T^.List_Id <> Nil
            Then
              Begin

                      a  :=  T^.a      ;
                      b  :=  T^.b      ;
                      P  :=  T^.List_Ld;

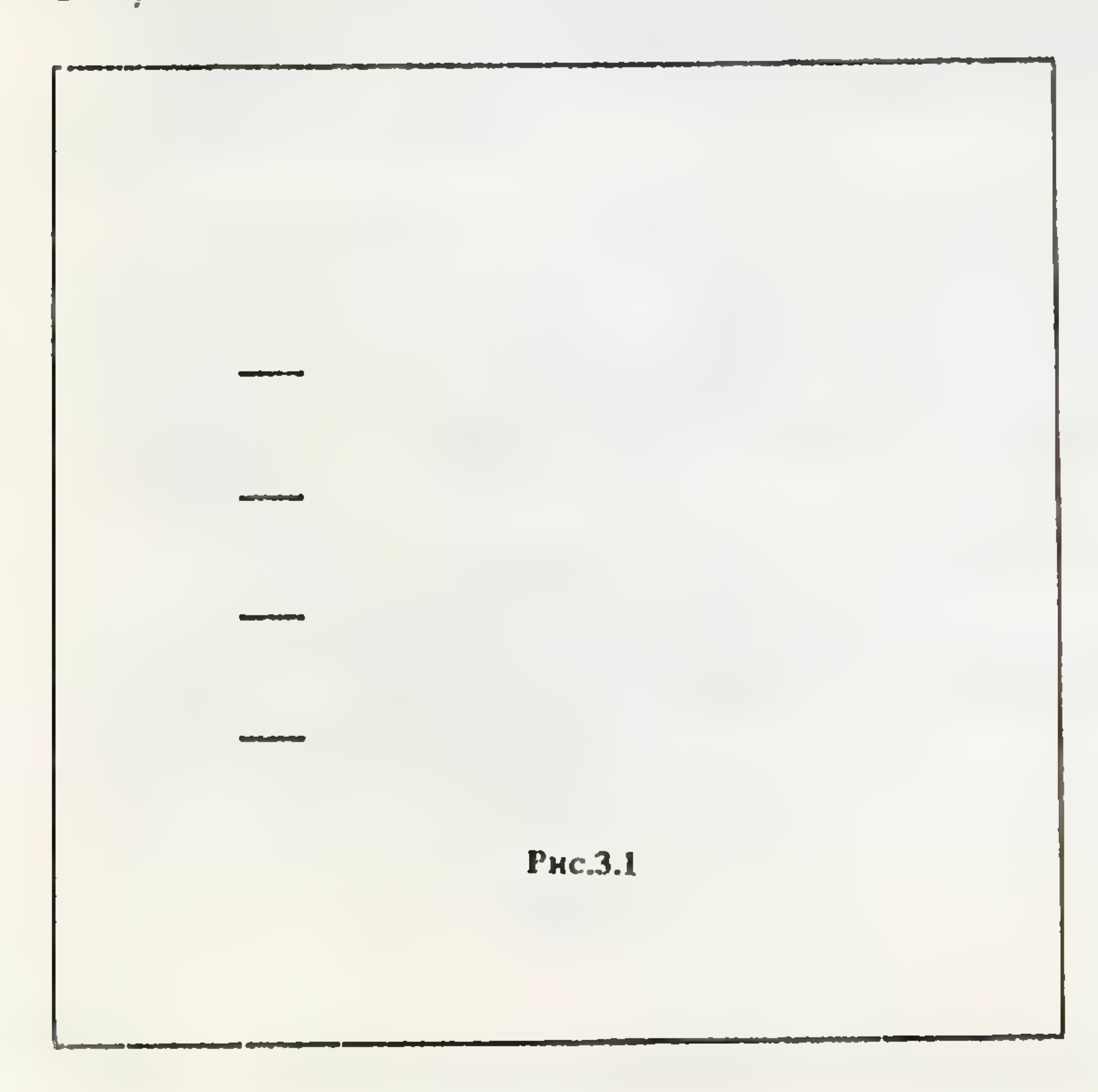
                      While p <> Nil
                        Do
                          Begin
                            Dist :=P^.Dist;
                            If Dist>0
                              Then {Ver}
                                Line(X0+Dist*ScX, Y0-a*ScY,
                                     X0+Dist*ScX,Y0-b*ScY)
                              Else {Hor}
                                Line(a*ScX+X0,Y0+Dist*ScY,
                                     b*ScX+X0,Y0+Dist*ScY);

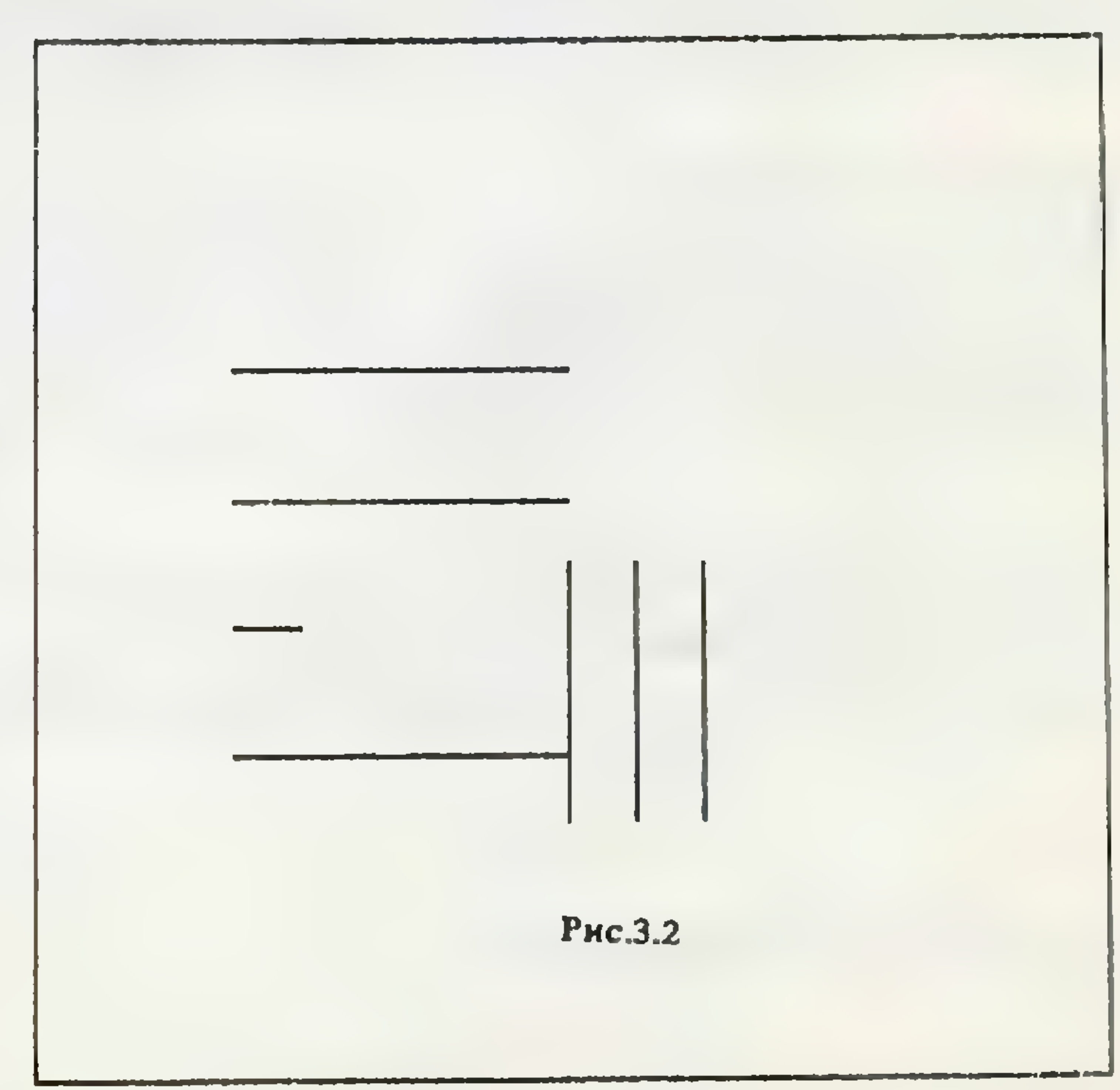
                          p  := p^.Ref   ;
                        End; s:=ReadKey;
              End;
        Scan_n_Draw(T^.LLink);
        Scan_n_Draw(T^.RLink);

    End;
End;
```

Сопоставим таблицу 2, содержащую список десяти стандартных отрезков, образуюших ПФ, и серию десяти рисунков 3.1 — 3.10.

На каждом рисунке показано включение в текущее изображение ПФ очередного стандартного отрезка из табл. 2 на указанных в списке расстояниях от осей ОХ, ОУ.


Рис.3.1


Рис.3.2

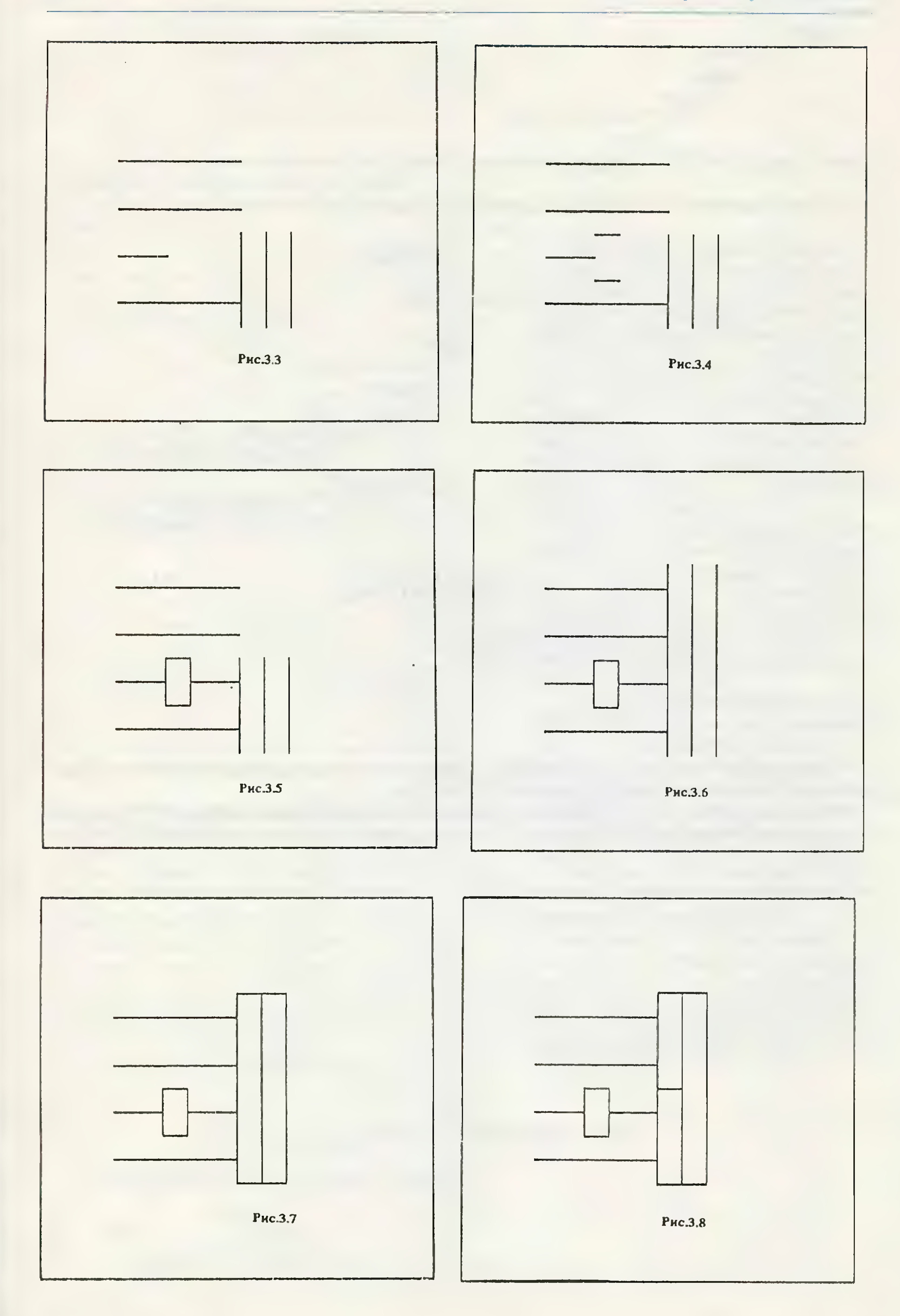


Рис.3.3

Рис.3.4

Рис.3.5

Рис.3.6

Рис.3.7

Рис.3.8

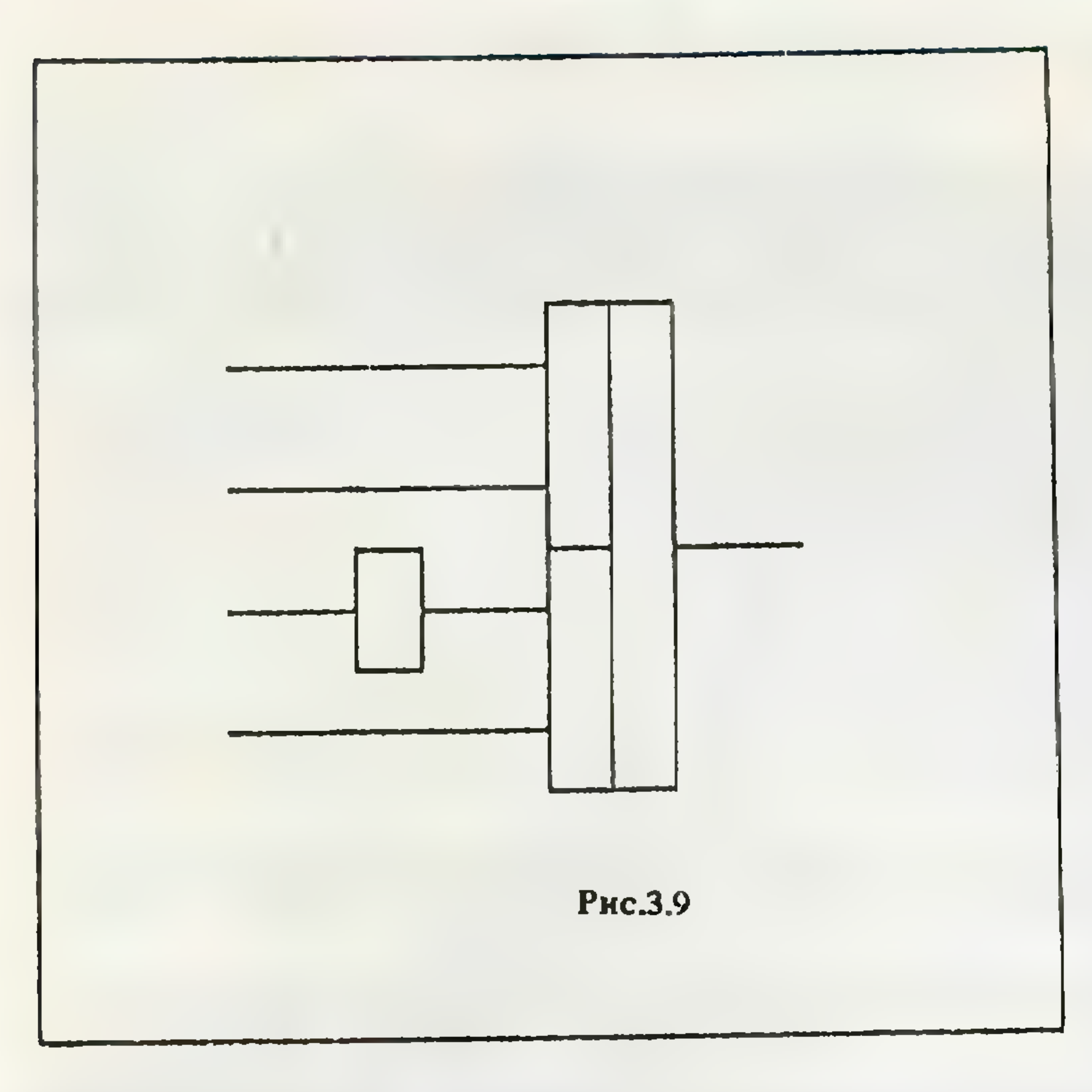


Рис.3.9

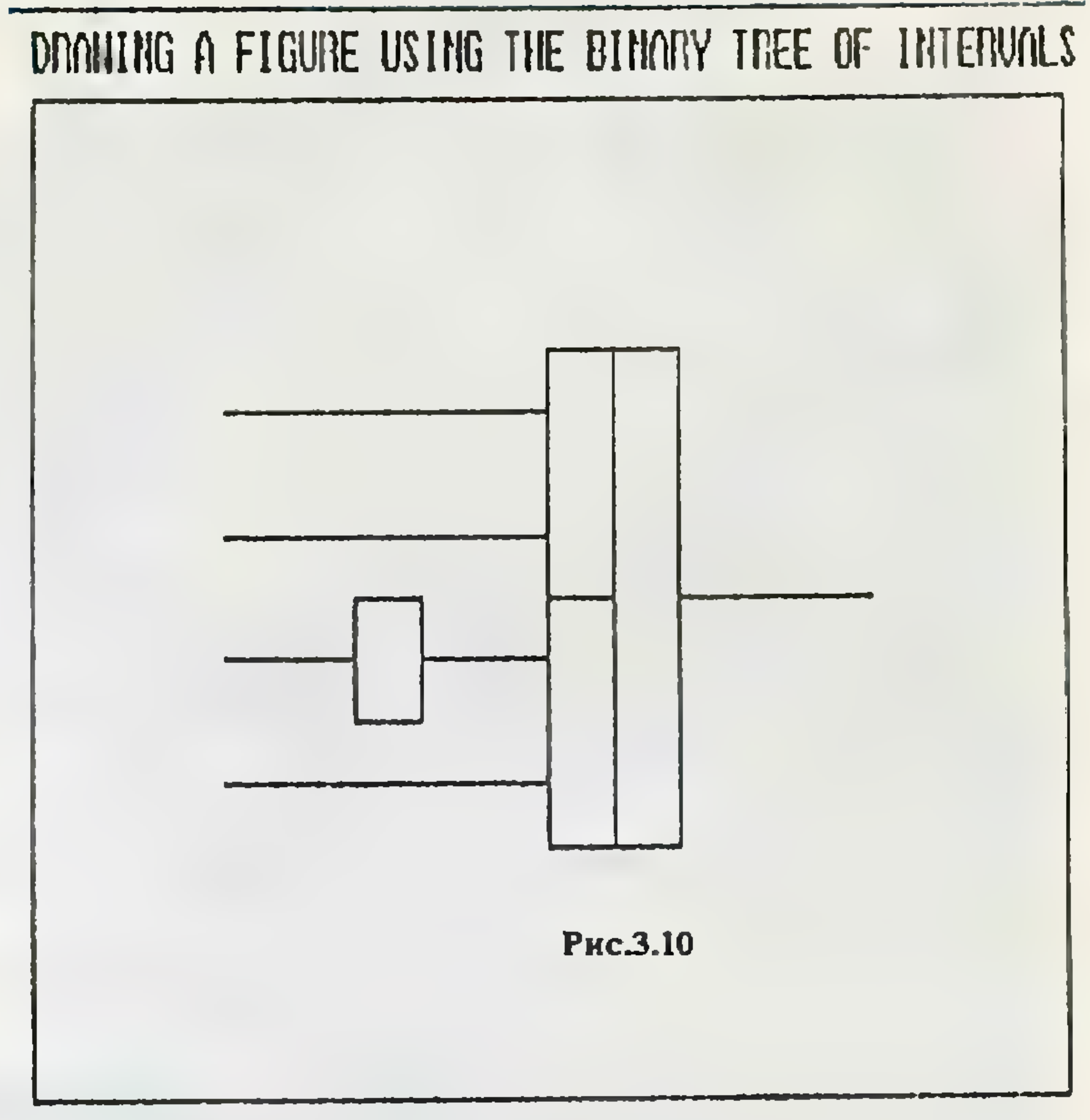


Рис.3.10

P.i.a., 1991

Комментарий специалиста

В работе рассматривается очень интересная структура данных: бинарное дерево отрезков, снабженное специальными "помечающими" линейными списками. Эти списки выделяют из общего количества отрезков бинарного дерева те из них, которые в совокупности образуют геометрический объект — прямоугольную фигуру.

Само бинарное дерево является статической структурой и строится один раз для данного универсального отрезка (стороны прямоугольного окна). Однако разметка дерева и, следовательно, представление фигуры — динамические. Построение и редактирование фигуры, т.е. задание и корректирование помечающих списков, осуществляется операциями Tick_Int и Del_Int, вставляющими и удаляющими отрезки, образующие фигуру.

Помимо основного преимущества указанной структуры, для представления геометрических объектов, т.е. возможности ее динамического изменения, имеются и некоторые другие.

В частности, без привлечения каких-либо дополнительных средств обеспечивается представление несвязных фигур, что при выборе некоторых других структур данных вызывает определенные трудности. В свою очередь, возможность задания несвязных фигур существенно облегчает реализацию операций конструктивной геометрии над прямоугольными фигурами (объединение, пересечение, разность).

Те, кто заинтересуется предложенной проблематикой, могут попробовать написать процедуру Del_Int, удаляющую отрезок фигуры, а также процедуры Union, InterSec и Diff, которые выполняют теоретико-множественные операции над прямоугольными фигурами.

Доцент И.А.Переход

Н.Б.Носкова

МАШИННАЯ ГРАФИКА: ПЛЮСЫ И МИНУСЫ

Графические дисплеи с их неограниченными возможностями синтеза изображения любого требуемого качества становятся сегодня такими же привычными средствами проектирования, как традиционные карандаши и циркули, а в некоторых областях даже заметно вытесняют последние. Сегодня уже трудно себе представить современную проектную фирму, абсолютно не использующую в своей практике хотя бы ограниченный спектр возможностей вычислительной техники.

Изобразительные возможности компьютеров широко рекламируются на страницах отечественной и зарубежной печати. В то же время главная задача всех публикуемых на эту тему материалов состоит в освещении аспектов проблемы, которые выгодно отличают рекламируемую систему или программу от существующих. Таким образом, уже сама постановка задачи предполагает ориентацию на анализ только положительных качеств машинной графики. В результате может возникнуть впечатление, что никаких недостатков вообще не существует, а если таковые и имеют место быть, то только вследствие несовершенства технического уровня и легко устраняются при соответствующем повышении последнего.

В то же время эффективное использование машинной графики в проектном процессе предполагает строгий учет объективных недостатков этого средства, которые далеко не всегда зависят только от технического уровня системы.

Достоинства вычислительной техники достаточно полно освещаются в профессиональной печати. К объективным преимуществам, выделяющим компьютерную графику в ряду других изобразительных средств, относятся:

- быстродействие, позволяющее в 3 — 4 раза быстрее получать изображение требуемого качества, а в конечном итоге в такой же пропорции сократить время на весь процесс проектирования за счет сокращения временных затрат на выполнение нетворческих операций;

- возможность имитации движения наблюдателя, а следовательно, и анализа проектируемого объекта в динамике восприятия;

- легкость взаимоперехода от ортогональной к пространственной проекции объекта, что особенно важно в процессе обучения архитектурному проектированию (рис.1);

- создание условий восприятия, приближающих изображение объекта к реальности (рис. 2).

Повышению натуралистичности изображения в последнее время уделяется особенно много внимания, так как именно этот аспект более других связан с решением целого круга технических задач, обеспечивающих не только качество, но и скорость выполнения определенных операций. Наиболее сложными среди них до сих пор считались введение цвета, полутонов, имитация эффектов естественного и искусственного освещения, обеспечение условий бинокулярного зрения (стереоскопический эффект) и т.п.

Сокращение временных и трудозатрат на получение высококачественного изображения сегодня достигается прежде всего за счет синтеза компьютерного рисунка проектируемого объекта и реальной фото-, кино- или видеосъемки окружающей среды, в том числе и природной. Аналитические возможности вычислительной техники позволяют в этих случаях точно и в считанные секунды совмещать ракурсы изображений, синхронно изменять их соответственно перемещению видовой точки и пр.

Отдавая должное широчайшим возможностям вычислительной техники, а также перспективам ее внедрения в проектный процесс, необходимо четко осознавать определенные ограничения



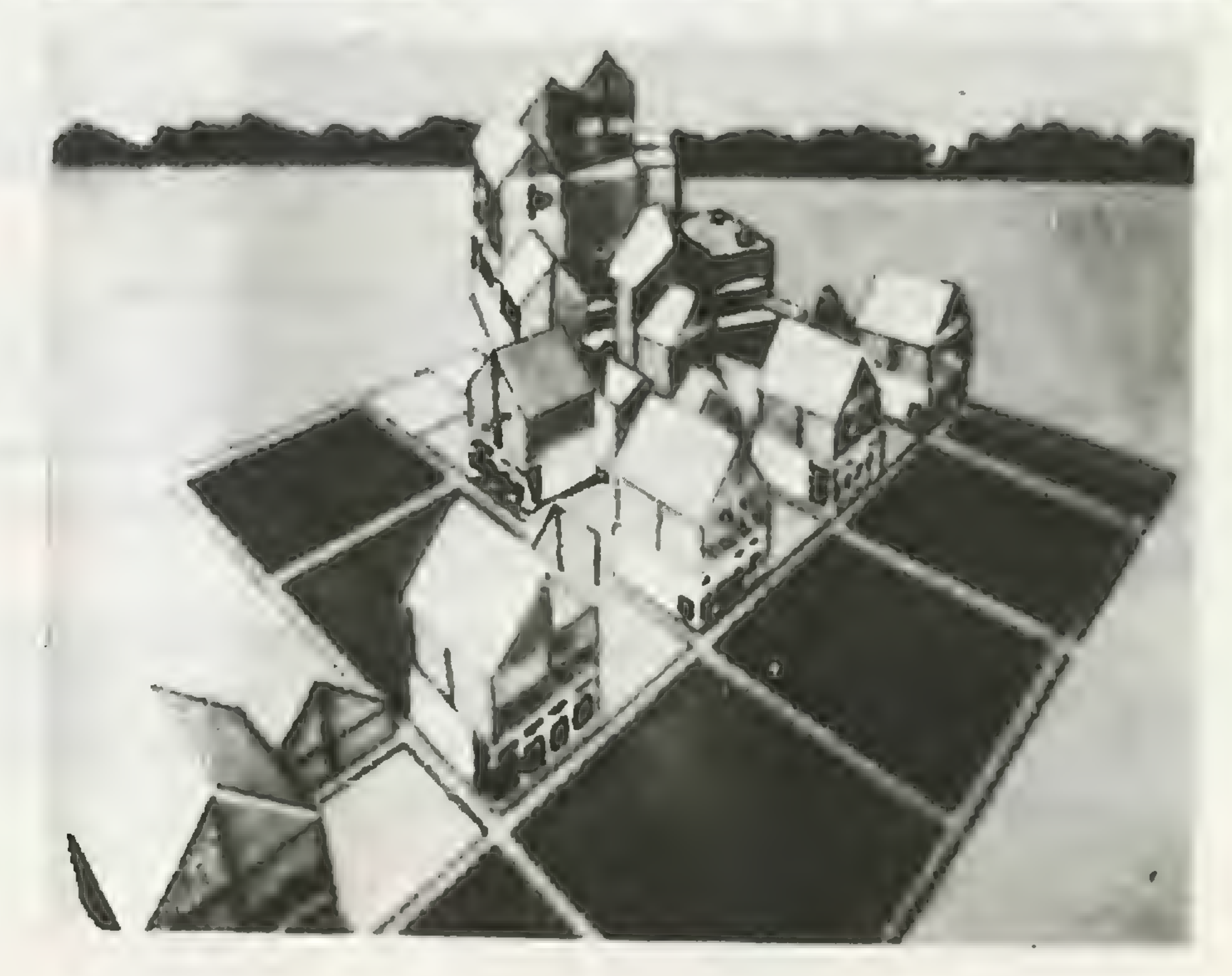


Рис.1



Рис.2. Справа — фотография фрагмента Новодевичьего монястыря; слева — перспективное изображение того же объекта, построенное традиционным способом на картинную плоскость, перпендикулярную основной оптической оси

использования этого средства, которые вытекают из некоторых особенностей получения изображений.

Прежде всего машинная графика не может быть использована в целях убеждения заказчика и потребителя в правильности предлагаемого решения, а также в качестве основы для проведения психологических, социальных и подобных экспериментов, предполагающих анализ оценочной реакции непрофессионального наблюдателя. В основе этого ограничения некорректность передачи информации за счет построения перспективного изображения любым из известных в начертательной геометрии способов. Характер и природа возникающих при этом искажений пространственных пропорций объекта исчерпывающе описаны в работах Ю.Королева и М.Федотова.

Перспективное изображение предполагает создание иллюзии пространства на двухмерной плоскости путем определенных геометрических построений. При этом хотя бы одно из трех изображаемых измерений пространственного объекта передается искаженно. Способ построения (на криволинейную картину, сферическую, цилиндрическую и пр.) может только в определенной степени уменьшить эти искажения, но не устранить их полностью.

Все существующие программы построения перспективных изображений на экранах дисплеев имеют в основе традиционный алгоритм (как правило, построение перспективы на перпендикулярную основному оптическому лучу картинную плоскость). Таким образом, машинная графика не устраняет свойственных традиционной перспективе искажений, а только усугубляет их собственными, специфическими, связанными с особенностями считывания информации и кривизной экрана дисплея.

Другая важная особенность перспективных искажений состоит в том, что полностью они могут быть выявлены только при сопоставлении реального и изображенного объектов. Таким образом, на основе перспективного изображения может сложиться ошибочное представление о пространственной структуре проектируемого объекта.

Важным недостатком, ограничивающим демонстрационные возможности компьютерной графики, является так называемый эффект технического шока, нередко возникающий у непрофессионального наблюдателя при демонстрации компьютерного мультфильма. Этот эффект возникает вследствие специфичности условий восприятия компьютерного изображения, которые, в свою очередь, обусловлены рассмотренными выше искажениями пространственных пропорций изображаемых объектов. Суть эффекта состоит в подсознательной переориентации внимания наблюдателя с оценки содержания проекта, т.е. того, что демонстрируется, на оценку качества изображения, т.е. того, как демонстрируется.

Второе ограничение использования машинной графики связано с корректностью принимаемых на основе перспективных изображений авторских решений и определяется степенью искажаемости изображений.

В отличие от непрофессионалов архитекторы обладают определенными навыками считывания и коррекции искажений. Эти навыки формируются еще в процессе обучения и совершенствуются по мере накопления опыта практической работы. В то же время некоторые способы построения перспективных изображений (широкоугольные перспективы, остроракурсные и т.п.) изображают пространственные про-

порции объекта настолько искаженно, что они вообще несопоставимы с реальными. Так, например, искажения пропорций объекта на изображения получены с разных точек.

Пределы допустимых искажений установлены опытным путем и определяются прежде всего величинами углов зрения: горизонтальный угол 30° — при двух точках схода и до 60° — при центральной перспективе. Вертикальный угол зрения — не менее 45°, что соответствует восприятию объекта с расстояния не менее одной высоты этого объекта.

Переводя эти пределы на масштабные уровни проектирования, устанавливают зависимость корректного решения на основе компьютерной визуальной модели. Такие решения могут приниматься только на объемных уровнях проектирования (здание, интерьер и т.п.), т.е. в масштабах 1:100 и крупнее. На градостроительных уровнях (комплекс, район и т.п.) возникает необходимость моделировать пространства, предполагающие построение широкоугольных перспектив (улица, площадь, панорама и т.п.) или остроракурсных изображений. При этом искажения настолько велики, что нельзя говорить о корректности решений, принимаемых на основе подобных изображений. Очевидно, что градостроительные уровни проектирования предполагают иные приемы визуального моделирования проектного предложения.

Еще одним существенным недостатком компьютерной графики, который, правда, в отличие от описанных выше все же в определенной степени связан с техническими особенностями системы, является относительность быстродействия получения изображения требуемого качества. Если в изображении встречается значительное количество повторяющихся элементов (типовые конструкции, типовые элементы рабочих чертежей и т.п.), то эффективность использования машинной техники для выполнения графических операций возрастает прямо пропорционально количеству повторов.

Если же речь идет о получении модели проектируемого объекта, или, говоря другими словами, перспективного изображения определенного качества, то эффективность использования вычислительной техники зависит от:

качества изображения (степень детализации, качество проработки деталей, степень сложности цветового решения, степень натуралистичности изображения и т.п.);

от количества видовых точек;

наличия или отсутствия элементов окружения.

Эффективность использования прямо пропорциональна количеству перечисленных элементов, или, говоря на языке проектирования, использование компьютеров для получения визуальных моделей очень эффективно при реконструкции. При незначительном количестве видовых точек и отсутствии опорной информации (сохраняемой застройки, элементов окружающей среды и т.п.) построение перспективного изображения на ЭВМ в некоторых случаях (в зависимости прежде всего от особенностей программного обеспечения) может занимать даже больше времени, чем традиционный ручной способ.

Получение различной архитектурно-проектной информации с помощью вычислительной техники является объективным отражением научно-технического прогресса. Доля вычислительной техники в общем объеме архитектурно-проектных работ неизбежно будет возрастать в силу уникальных возможностей этого изобразительного средства, выгодно отличающих его от всех остальных. Однако эффективность использования этого наиболее совершенного средства в проектировании предполагает и знание объективных ограничений его использования. Ряд особенностей получения информации позволяет утверждать, что использование компьютерной графики для получения высокоиллюзорных симуляций совершенно нерационально. В то же время использование аналитических возможностей ЭВМ, а также изобразительных на стадии рабочего проектирования как средства механизации нетворческих процессов будет все более эффективным по мере повышения качества программного обеспечения и совершенствования технического уровня систем.

АНАЛИЗ ИСКАЖЕНИЙ ПЕРСПЕКТИВНОГО ИЗОБРАЖЕНИЯ, ПОСТРОЕННОГО НА ЭВМ

Получение различной архитектурно-проектной информации с помощью вычислительной техники является объективным отражением современного научно-технического прогресса. Технические средства и многочисленные программы к ним широко рекламируются в зарубежных журналах, проектные институты стремятся найти валюту на их приобретение.

К сожалению, уникальные возможности вычислительной техники нередко рассматриваются сегодня в качестве своеобразной панацеи от всех бед архитектурного проектирования. При этом потребитель зачастую ожидает от совершенного технического инструмента не только выполнения самых сложных и трудоемких операций, но и полноценного участия в творческом процессе на равных с человеком.

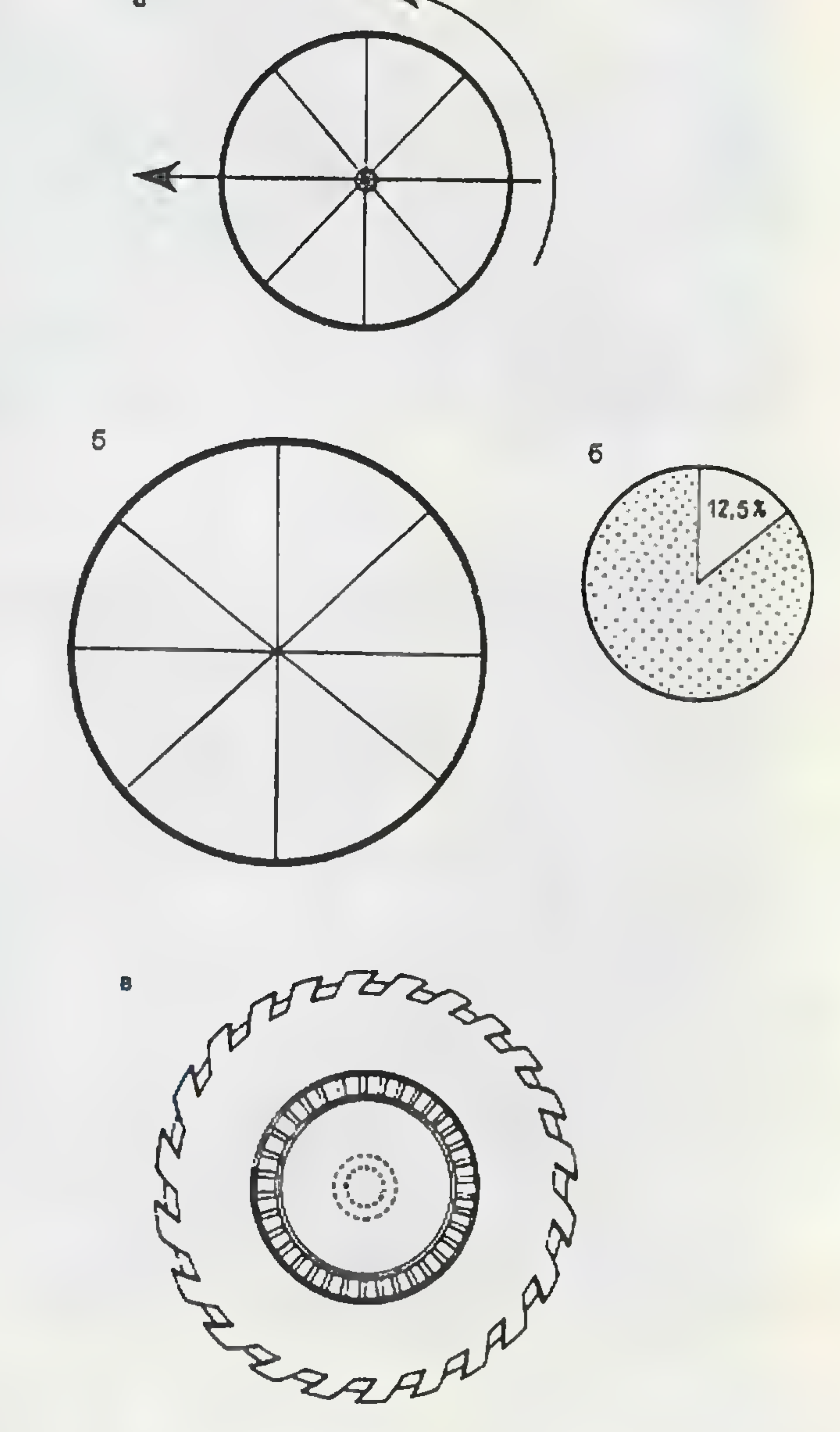


Рис.3

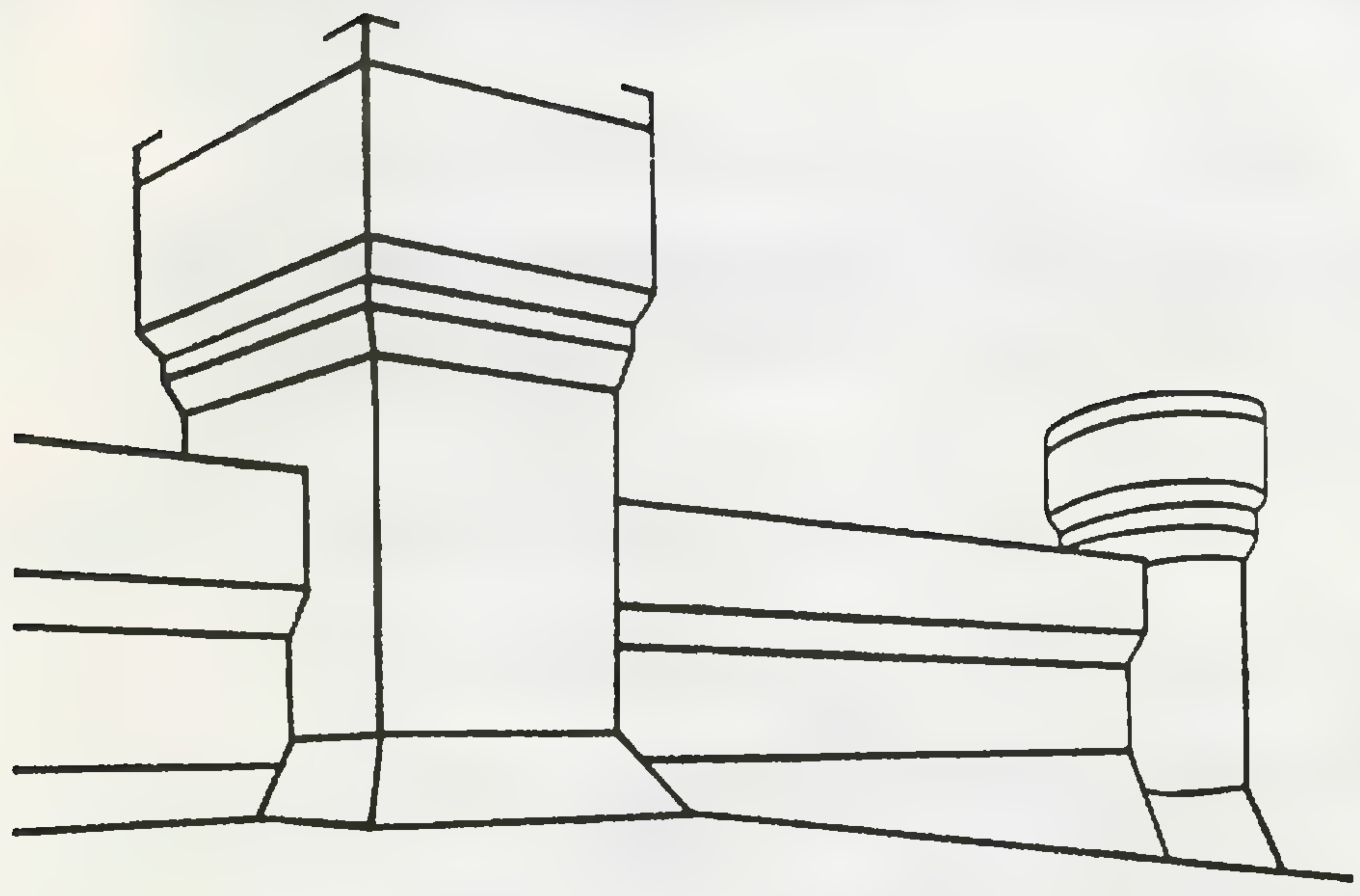


Рис.4. Фотография фрагмента Новодевичьего монастыря (вверху) и схема силуэта объектов по фотографии (внизу)

Отдавая должное широчайшим возможностям вычислительной техники и перспективам ее внедрения в проектный процесс, необходимо четко осознавать определенные ограничения использования этого средства в проектировании, в частности для демонстрации проектной идеи заказчику, потребителю или эксперту с целью оценки проекта, а также в качестве основы для проведения социальных, психологических и других экспериментов.

Рассматривая возможности графической информации адекватно отражать авторскую проектную идею, необходимо в первую очередь оговорить и сформулировать условия, принимаемые в качестве критерия корректного изображения проектной идеи.

Известная степень условности любого изображения создает предпосылки возникновения разночтения изложенной таким способом информации. Это можно проиллюстрировать на конкретных примерах.

Интересный эксперимент проводил американский художник-график У.Боумен. Он предложил математику, экономисту и автомобилисту изобразить хорошо знакомый всем объект — колесо. При этом главная задача состояла в "отражении идеи формы". Анализируя полученные изображения, Боумен обнаружил, что рисунок математика больше напоминал схему вращения тела вокруг собственной оси (рис.3,а), экономиста — соотношение абстрактных стоимостей (рис.3,б), автомобилиста — конструктивный чер-

теж (рис.3,в). Интересно, что определить по рисунку, какой предмет изобразили его коллеги по эксперименту, ни один из участников не смог.

Аналогичные эксперименты проводились и другими исследователями. В результате было установлено, что, для того чтобы по изображению судить о качестве объекта, наблюдатель должен иметь возможность сопоставить условия восприятия этого изображения и аналогичного объекта в реальной среде. При этом желательно, чтобы условия восприятия изображения и натуры максимально приближались друг к другу.

Таким образом, можно утверждать, что психологами экспериментально установлены следующие условия корректного изображения авторской проектной идеи:

- приближение условий восприятия изображения к натуре;
- устранение разночтений графической информации, предполагающее соответствие качества демонстрируемого изображения уровню подготовленности наблюдателя к восприятию представленной таким способом информации, в том числе и автоматической, т.е. в уме, коррекции присущих способу искажений.

В качестве эталона корректного изображения качественных характеристик объекта в этих же экспериментах приводится фотография объекта. Широкое распространение этого способа практически во всех областях человеческой деятельности выполняет роль специальной подготовки, обеспечивая автоматическую коррекцию зрителям свойственных этому способу искажений. В то же время сами искажения фотоснимка, возникающие вследствие особенностей работы оптической системы, весьма специфичны, что также способствует их выявлению и коррекции.

С другой стороны, сфотографировать можно только существующий объект. В проектной же практике мы имеем дело с несуществующим объектом, поэтому критерием корректности изображения в этом случае является приближение условий восприятия изображения к фотографии. К таким способам наглядного представления информации относятся прежде всего перспективные изображения, рассматриваемые в данной статье, и макетоскопические изображения, представляющие материал для отдельной темы.

Перспективное изображение позволяет путем определенных геометрических построений создать на двухмерной плоскости листа (или экрана дисплея) иллюзию трехмерного пространства и отразить таким образом закономерности изменения восприятия пропорций объекта при перемещении его относительно других объектов и наблюдателя. В отличие от обычного фотоснимка искажения такого изображения возникают вследствие "идеологии" самого способа и носят "скрытный" характер. В большинстве случаев некорректность изображения пропорций объекта может быть обнаружена только при сопоставлении изображения и натуры (рис.4).

Если в традиционном варианте построения перспективного изображения проектировщик имеет возможность варьировать вертикальный и горизонтальный углы зрения, изменяя таким образом границы видового кадра, то в машинном варианте условия работы программы, как правило, предполагают пропорциональную зависимость между этими углами. Таким образом, изменение вертикального угла зрения возможно только при соответствующем изменении горизонтального, что на практике соответствует удалению или приближению точки зрения к заданному положению картинной плоскости.

В конкретном варианте такие перемещения приводят к тому, что требуемый видовой кадр (рис.5) получен машиной с несуществующей видовой точки. С заданного же положения наблюдателя, т.е. видовой точки, с которой построено традиционное изображение и сделана фотография, в машинном варианте в кадр попадает толь-

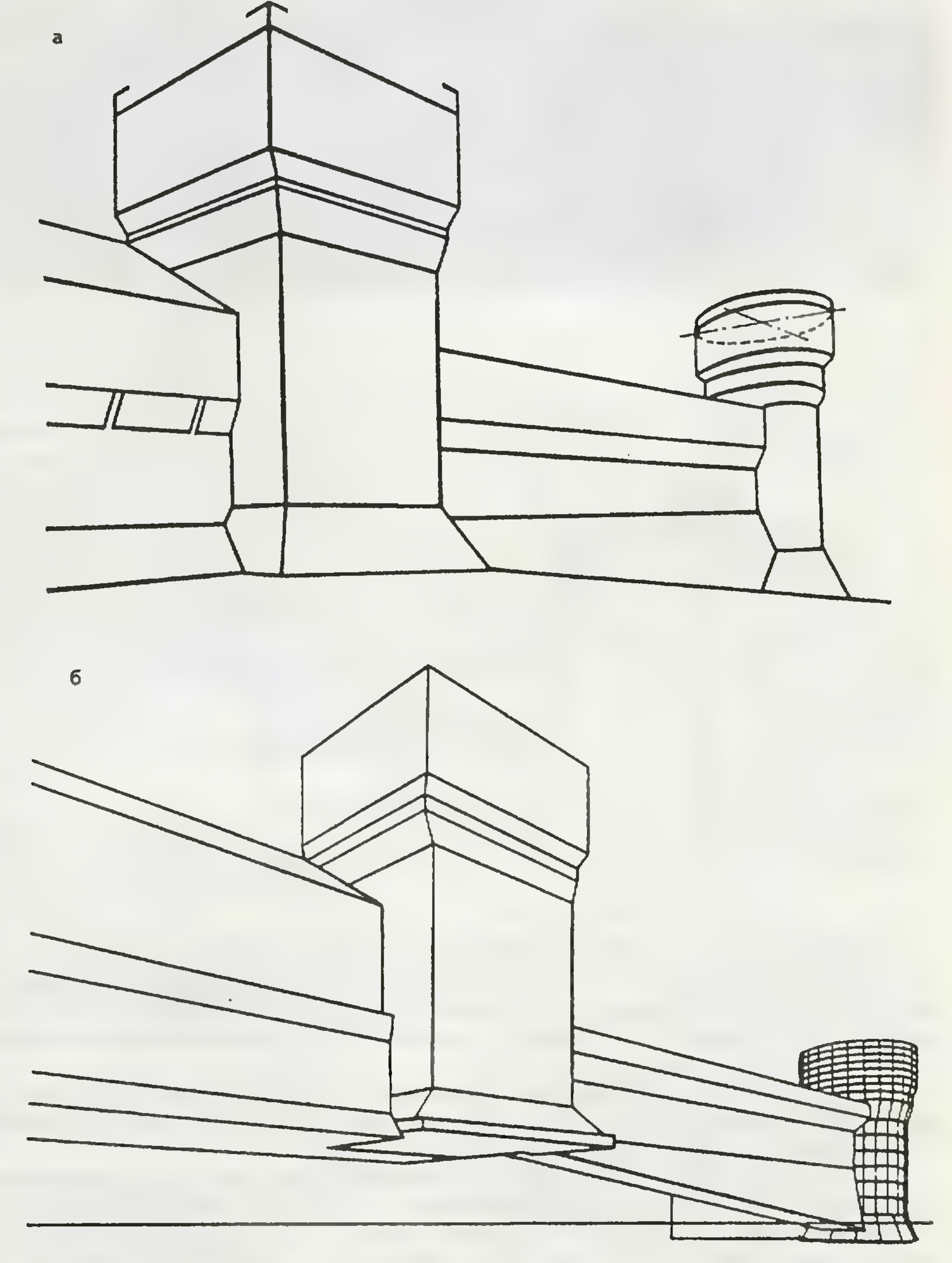


Рис.5. Сравнение перспективного изображения фрагмента Новодевичьего монастыря в Москве, выполненного традиционным способом (а) и при помощи ЭВМ (б)

ко фрагмент стены. Пропорциональное уменьшение видового кадра приводит к тому, что реальная точка зрения как бы удаляется, а соответственно изменяется и искажаемость пропорций объектов, становясь более соответствующей восприятию с другой видовой точки.

Другой особенностью всех перспектив, получаемых машинным способом, является специфичность их восприятия, особенно если речь идет о машинном мультфильме, имитирующем движения наблюдателя или его головы или глаз. В этом случае у зрителя, особенно не обладающего определенной профессиональной подготовкой, возникает так называемый эффект технического шока, когда внимание наблюдателя помимо его воли приковывается к тому, каким образом получено изображение, а не к качеству демонстрируемого содержания.

Таким образом, исходя из выше сказанного, можно предположить, что использование вычислительной техники как аналитического инструмента оценки правильности предполагаемого проектного решения, а также демонстрации потребителю проекта с целью получения оценочной реакции дает заведомо некорректный результат. Другое дело — использование машинной графики как средства самоконтроля проектировщика. Однако и в этом случае трудоемкость операций ввода и вывода информации, являющейся основным камнем преткновения на пути современного внедрения вычислительной техники в проектный процесс, позволяет предположить, что использование вычислительных систем для выполнения высокоиллюзорных симуляций совершенно нерационально. Высокоэффективными аспектами использования ЭВМ по-прежнему остаются их аналитические возможности, а также получение ортогональных проекций, особенно на стадии рабочего проектирования.

Вместе с тем следует надеяться, что дальнейшее совершенствование вычислительной техники наряду с разработкой более эффективной методики ее использования в архитектуре позволит повысить качество получаемой информации, в том числе в решении тех проблем, о которых говорилось выше.

Д.Ю.Усенков

Электронное зеркало

Один из интересных эффектов, который можно применить в программах на Бейсике, — превращение картинки на экране в ее зеркальное отображение. Этого можно легко добиться, перебирая все точки построчно по их координатам в цикле и изменяя соответственно их цвета с помощью операторов PSET и POINT. Но для работы такой программы требуется довольно много времени — сказывается низкое быстродействие Бейсика (и это на специально созданного для БК-0010.01 Быстрого Бейсика, построенного по типу компилятора; сколько же времени потребовалось бы для работы интерпретирующего языка!). Безуспешно попытавшись ускорить процесс путем оптимизации Бейсик-программы, я составил подпрограмму в машинных кодах. Кроме того, в ней я вообще отказался от алгоритма перебора по координатам и замены цвета (это потребовало бы составления программы-аналога POINT). Вместо этого я использовал алгоритм работы с ячейками экранного ОЗУ: производится последовательный построчный перебор пар ячеек, расположенных симметрично относительно середины экрана; выбранные значения помещаются в регистры и преобразуются в свои зеркальные отражения при помощи несложного алгоритма:

MOV #10,R0   в R0 записано число битов в байте
1$: RORB R4
ROL B R1 в R1 после выполнения этой последовательности команд находится зеркальная копия R4
SOB R0, 1$

После этого преобразованные значения меняем местами и возвращаем в выбранные ячейки памяти. При работе этой подпрограммы зеркально отражается верхняя, информационная строка и меняется цвет точек синий на зеленый и обратно. Вот текст в восьмеричных кодах.

```
Ø1ØØ46  Ø1Ø146  Ø1Ø246  Ø1Ø346  Ø1Ø446  Ø127Ø4  Ø777ØØ  Ø127Ø3
ØØØØ37  Ø1Ø346  Ø1Ø446  Ø616Ø3  Ø1Ø346  1113Ø3  Ø127ØØ  ØØØØ1Ø
1Ø6ØØ3  1Ø61Ø2  Ø77ØØ3  Ø126Ø3  Ø627Ø4  ØØØØ77  1666Ø4  ØØØØØ2
Ø1Ø446  1114Ø4  Ø127ØØ  ØØØØ1Ø  1Ø6ØØ4  1Ø61Ø1  Ø77ØØ3  Ø126Ø4
11Ø113  11Ø214  Ø126Ø4  Ø126Ø3  ØØ53Ø3  ØØ2343  1627Ø4  ØØØ1ØØ
Ø2Ø427  Ø4ØØØØ  ØØ2334  Ø126Ø4  Ø126Ø3  Ø126Ø2  Ø126Ø1  Ø126ØØ
ØØØ2Ø7  ØØØØØØ  ØØØØØØ  ØØØØØØ
```

Длина: 150$_8$. Начальный адрес может быть любым, например 37000$_8$. Контрольная сумма: 133114$_8$.

Эта подпрограмма без всяких изменений может быть использована и в кодах и даже в Фокале, если найдется способ ее вызвать. Ввести ее можно либо с помощью операторов POKE в цикле, либо в режиме ТС (после этого ее надо записать на магнитофон с именем, допустимым в Бейсике, например, "ЗЕРКАЛО.BIN).

Хочу дополнить информацию, помещенную в "Вычислительной технике" № 5 за 1990 г., стр.34. В статье Синягина С.Ю. он рекомендует работать с системами МИРАЖ и МИКРО. Ничего не имея против них, особенно против системы МИРАЖ, работающей в ОЗУ экрана и оставляющей программисту всю оперативную память, я хочу порекомендовать вашему вниманию весьма распространенную среди давно программирующих БК пользователей (у одного из них мне и удалось ее списать рижскую программу ОТЛАДЧИК. Фактически это микроассемблер (дизассемблер с возможностью отладки и пошагового прохождения программ.

Программа ОТЛАДЧИК занимает 10 кБ памяти, что все же позволяет создавать довольно большие программы.

Заключение рецензента

Подпрограмма работоспособна и может быть опубликована. Читатели могут попробовать написать аналогичные программы для зеркального переворота изображения относительно обеих диагоналей экрана и по вертикали. Подобные программы могут быть полезны для создания экранных эффектов в игровых программах.

Жариков Л.Н.

Н.Арбузов

Совет

Недавно прочитал в вашем журнале (8/1990) в статье Д.Ю.Усенкова "О некоторых периферийных устройствах для БК 0010" об устройстве ввода УВК-1 (мышь) и решил купить. Купил, подключил, набрал программу, приводимую в инструкции по эксплуатации, — мышь работает. Но как к ней обращаться программно, в инструкции не написано. Надо самим разбираться в приведенной программе. Для тех, кто знает коды, это просто. А для тех, кто не знает? Что им делать?

Чтобы они не мучались, хочу дать совет: Надо сначала инициировать порт ввода-вывода, заслав туда число, скажем, десять (в восьмеричной), только затем в ячейке с адресом 177714_8 появляются коды поступающих от "мыши" сигналов. Привожу программу обращения к УВК.

```
10 X%-100%' задание начальной координаты по X
20 У%-100%' задание начальной координаты по У
30 C%-1%' задание начального цвета
40 P%-&O177714' инициализация
50 L%-PEEK(P%)' порта
60 POKE P%,0%' ввода
70 POKE P%' &O10' вывода
80 IF L%-&O1 THEN У%-У%-1%' определения
90 IF L%-&O4 THEN У%-Y%+1%' кода
100 IF L%-&O10 THEN X%-X%-1%' поступившего
110 IF L%-&O2 THEN X%-X%+1%' от
120 IF L%-&O40 THEN C%-1%' мыши
130 IF L%+&O100 THEN C%-0%
140 IF L%-&O140 THEN END
150 PSET (X%, У%), C%' установка точки
160 PSET (X%, У%), C%' с координатами X, У
170 GOTO 50
```

Тут же привожу таблицу кодов, поступающих от мыши

Манипуляция мыши	Код на порту ввода-вывода
Вверх	1
Вниз	4
Вправо	2
Влево	10_8
Клавиша 'Л'	40_8
Клавиша 'П'	100_8

Все остальные операции являются суммой предыдущих, например, при одновременном нажатии клавиш 'Л' и 'П' появляется код 140_8, и т.д.

Так же, пользуясь случаем, хочу сказать несколько слов о спрайтах на БЕЙСИКе. В журнале ИНФО (2/1990) говорилось о спрайтах, но там написано, как их помещать в память, а как сделать так, чтобы они перемещались по экрану и не стирали изображения?

Ниже привожу программу, которая позволяет перемещать спрайты размером 8x8 с помощью клавиш управления курсором, не стирая при этом изображения нарисованного до этого.

Конечно, эту программу можно было бы уменьшить, заносить спрайт не прямо непосредственно в программу, а в оператор DATA или в какую-нибудь область ОЗУ, но для простоты и большего быстродействия привожу ее не сокращенной.

Эту программу можно переделать на мышь, изменив и вставив некоторые строки...

```
70  k%-peek(&O177714)
71 Poke &O177714,0%
72 Poke &O177714,8%
```

Вот эта программа:

```
10 PRINT CHR$ (140%)+CHR$ (140%)' УСТАНОВКА ЭКРАНА.
20 POR Y%-1 TO 20
30 FOR X%-1 26 STER 6
40 PRINT AT (X%,Y%); "SPRITE;"
50 NEXT X%
60 NEXT Y%
```

```
70 K%=PEEK (&O177662)
80 IF K%=8% THEN I%=I%-2%
90 IF K%=25% THEN I%=I%+2                    опрос клавиатуры
100 IF K%=26% THEN I%=I%-128%
110 IF K%=27% THEN I%=I%+128%
120 IF &H5000+I%<&H400  THEN I%=I%+128% '  предохранение от
130 IF &H5000+I%>&H7E00  THEN I%-I%-128% '  наездов спрайта на ОЗУ
140 A1%=PEEK(&H5000+I%)
150 A2%=PEEK(&H5040+I%
160 A3%=PEEK(&H5080+I%)          запоминание предыдущего
170 A4%=PEEK(&H5000+I%)
180 A5%=PEEK(&H5100+I%)          изображения
190 A6%=PEEK(&H5040+I%)
200 A7%=PEEK(&H5080+I)
210 POKE &H5000+I%,&HAEAE
220 POKE &H5040+I%,&HABAE
230 POKE &H5080+I%,&HAAEE
240 POKE &H50C0+I%,&HAABE   запись спрайта на это место
250 POKE &H5100+I%,&HAAEE
260 POKE &H5040+I%,&HABAE
270 POKE &H5080+I%,&HAEAE
280 FOR M=1 TO 20 'пустой
290 NEXT M                       цикл
300 POKE &H5000+I%,&B00
310 POKE &H5040+I%,&B00
320 POKE &H5080+I%,&B00          стирание спрайта
330 POKE &H50C0+I%,&B00
340 POKE &H5100+I%,&B00
350 POKE &H5140+I%,&B00
360 POKE &H5180+I%,&B00
370 POKE &H5000+I%,A1%
380 POKE &H5040+I%,A2%
390 POKE &H5080+I%,A3%           запись изображения
400 POKE &H50C0+I%,A4%
410 POKE &H5100+I%,A5%
420 POKE &H5140+I%,A6%
430 POKE &H5180+I%,A7%
440 GOTO 70
```

Хочу поделиться полезным советом: чтобы лучше работала клавиатура, надо ее смазать машинным маслом, после чего кнопки реже заедают, и чтобы с клавиш не стиралось изображение символов, их надо покрыть лаком, желательно мебельным.

Заключение рецензента

Предлагаемые программы представляют интерес для начинающего программиста.

Хотя имеются программы достаточно высокого уровня в кодах для обращения к "мыши" и для создания движущихся без стирания фона спрайтов (например, в статье О.Д.Любутова, серия ВТ 1/1991, с. 33), но для начинающего программиста предлагаемые простые программы на БЕЙСИКе полезны и поучительны. Желающие могут доработать вторую программу с тем, чтобы получить шаг перемещения спрайта не на ширину курсора, а на одну точку.

Ю.В.Кузьмин



Как быть, если одной персональной ЭВМ недостаточно, а цены на локальные сети кажутся непомерно высокими?

МЫ РЕШИМ ЭТУ ЗАДАЧУ ДЛЯ ВАС!

Львовское МП "Инвестсервис" предлагает

МНОГОТЕРМИНАЛЬНЫЙ КОМПЛЕКС КРАБ НА БАЗЕ ПЭВМ IBM PC AT/XT.

Комплекс КРАБ - это:

- полная совместимость с системами MS/DOS, PICK, PC/MOS, XENIX, MSM...
- комплектация лучшими отечественными терминалами, возможно подключение собственных аппаратных средств заказчика;
- многоканальные адаптеры, обеспечивающие подключение 4 — 16 терминалов.
Неплохо, правда? Но главный сюрприз впереди, когда вы обратитесь за дополнительной информацией и узнаете смехотворно низкую цену и заманчивые условия поставки.
КРАБ успешно применяют медики и нефтяники, филологи и шахтеры.
КРАБ покупают все - от крохотного кооператива до гигантского объединения.
Разработчики программного обеспечения, купившие КРАБ, получают право поставлять его пользователям вместе с собственными программными изделиями и вступить в ассоциацию "KRABUS".

ЗАПОМНИТЕ - ПРОФЕССИОНАЛЫ ВЫБИРАЮТ КРАБ!

Наш почтовый адрес:
290044, Львов-44, а/я 8863
МП "Инвестсервис"

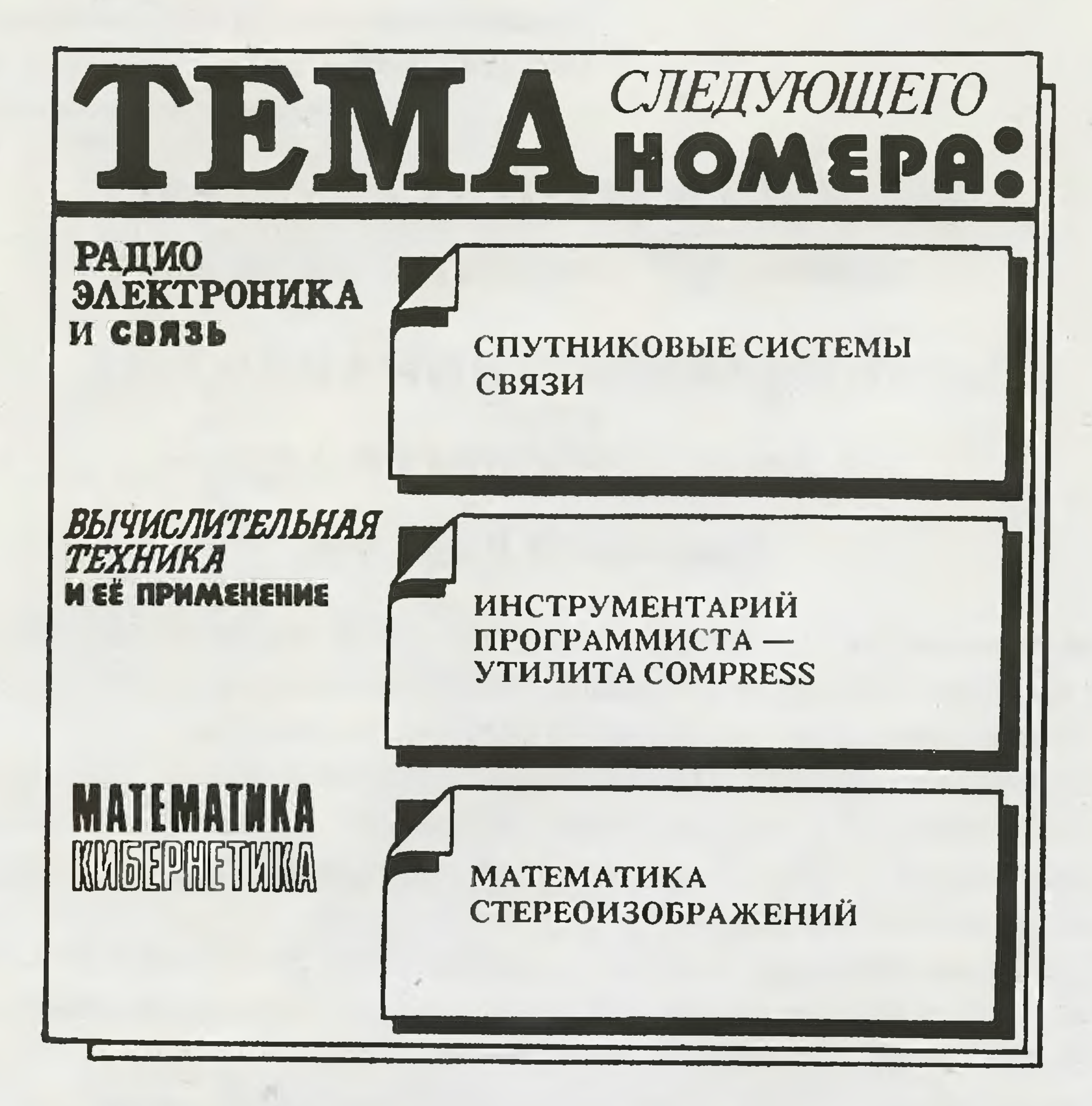
Контактные телефоны:
35-35-79, 34-32-12 с 8 до 17 часов;
34-29-42 круглосуточно.

ИНВЕСТСЕРВИС

Г 27 Графика в байтах. — М.: Знание, 1991. — 48 с. — (Новое в жизни, науке, технике. Сер. "Вычислительная техника и ее применение"; № 10).
ISBN 5-07-002202-4
35 к.

Очередной выпуск по теме: "Машинная графика и геометрия" охватывает следующие вопросы: пакет AUTUCAD, графика в архитектуре и др.
Материал рассчитан на широкий круг читателей.

2404040000 ББК 32.97



Научно-популярное издание

ГРАФИКА В БАЙТАХ

Зам. главного редактора *Г. Г. Карвовский*
Редактор *Б. М. Васильев*
Мл.редактор *Н. А. Васильева*
Художник *В. Н. Конюхов*
Худож.редактор *И. А. Емельянова*
Техн.редактор *Т. В. Луговская*
Корректор *В. И. Гуляева*

ИБ № 11868

Подписано к печати 13.08.91. Формат бумаги 70x100 1/16. Бумага офсетная. Печать офсетная. Усл.печ.л.3,90. Усл.кр.-отт.8,45. Уч.-изд.л.3,00. Тираж 50443 экз. Заказ 2424. Цена 35 коп. Издательство "Знание". 101835, ГСП, Москва, Центр, проезд Серова,д.4. Индекс заказа 9147010. Отпечатано с оригинал-макета, подготовленного издательством "Знание" в издательской системе Xerox Ventura Publisher (ОС MS DOS), на ордена Трудового Красного Знамени Тверском полиграфическом комбинате Государственного комитета СССР по печати. 170024, г.Тверь, пр.Ленина,5.

Цена 35 коп. Индекс 70195

Адрес подписчика:
Сол. 5-27



Подписная научно-популярная серия
ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА
И ЕЁ ПРИМЕНЕНИЕ

Весь наблюдаемый мир — это просто склад образов и знаков.

Бодлер

Нажатие одной клавиши на клавиатуре терминала может привести к 10 обращениям к отдельным программам операционной системы, к выполнению 1000 машинных команд и к 10 миллионам изменений состояний логических вентилей.

Питер Деннинг, Роберт Браун

Наш адрес:
101835,
Москва,
Центр,
проезд
Серова, 4